z/OS C/C++

IBM Open Class Library Reference

SC09-4812-01

z/OS C/C++

IBM Open Class Library Reference

SC09-4812-01

Note!
Before using this information and the product it supports, be sure to read the general information under "Notices" on page 3111.

Second Edition (October 2001)

This edition applies to Version 1, Release 2, Modification Level 0, of IBM® z/OS C/C++ (5694-A01), IBM Open Class™ Library, and to all subsequent releases and modifications until otherwise indicated in new editions.

This edition replaces SC09-2364-04.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below. You can also browse the books on the World Wide Web by clicking on ″The Library″ link on the z/OS home page. The web address for this page is
`http://www.ibm.com/servers/eserver/zseries/zos/bkserv`.

IBM welcomes your comments. You can send your comments to:

IBM Canada Ltd. Laboratory
Information Development
B3/KB7/8200/MKM
8200 Warden Avenue
Markham, Ontario, Canada L6G 1C7

You can send your comments electronically to IBM. See "How to Send Your Comments" for a description of the methods.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2001. All rights reserved.
US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

About This Reference lix

Who Should Use This Book lix
How to Use This Book lix
A Note about Examples lix
z/OS C/C++ and Related Publications lix
How to Send Your Comments lx

Chapter 1. Application Control Classes 1

IAttribute 1
IAttribute - Hierarchy List 1
IAttribute - Member Functions and Data by Group 1
IAttribute - Inherited Member Functions and Data 1
IAttributeName 2
IAttributeName - Hierarchy List 2
IAttributeName - Member Functions and Data by Group 2
IAttributeName - Inherited Member Functions and Data 3
ICondition 3
ICondition - Hierarchy List 4
ICondition - Member Functions and Data by Group 4
ICondition - Inherited Member Functions and Data 5
ICountedPointerTo 6
ICountedPointerTo - Hierarchy List 6
ICountedPointerTo - Member Functions and Data by Group 6
ICountedPointerTo - Inherited Member Functions and Data 10
ICritSec 10
ICritSec - Hierarchy List 10
ICritSec - Member Functions and Data by Group 11
ICritSec - Inherited Member Functions and Data 11
ICrossThreadNotificationLoop 11
ICrossThreadNotificationLoop - Hierarchy List 12
ICrossThreadNotificationLoop - Member Functions and Data by Group 12
ICrossThreadNotificationLoop - Inherited Member Functions and Data 13
ICurrentNonGUIApplication 13
ICurrentNonGUIApplication - Hierarchy List . . 13
ICurrentNonGUIApplication - Member Functions and Data by Group 13
ICurrentNonGUIApplication - Inherited Member Functions and Data 15
ICurrentNonGUIThread 15
ICurrentNonGUIThread - Hierarchy List 16
ICurrentNonGUIThread - Member Functions and Data by Group 16
ICurrentNonGUIThread - Inherited Member Functions and Data 18
IDLLModule 19
IDLLModule - Hierarchy List 19
IDLLModule - Member Functions and Data by Group 19
IDLLModule - Enumerations 23
IDLLModule - Inherited Member Functions and Data 24
IDynamicLinkLibrary 24
IDynamicLinkLibrary - Hierarchy List 24
IDynamicLinkLibrary - Member Functions and Data by Group 24
IDynamicLinkLibrary - Enumerations 29
IDynamicLinkLibrary - Inherited Member Functions and Data 29
IEnumHandle 30
IEnumHandle - Hierarchy List 30
IEnumHandle - Member Functions and Data by Group 30
IEnumHandle - Type Definitions 31
IEnumHandle - Associated Globals 31
IEnumHandle - Inherited Member Functions and Data 31
IEnvironment 32
IEnvironment - Hierarchy List 32
IEnvironment - Member Functions and Data by Group 32
IEnvironment - Inherited Member Functions and Data 35
IEventData 35
IEventData - Hierarchy List 36
IEventData - Member Functions and Data by Group 36
IEventData - Inherited Member Functions and Data 42
IExternalProcess 43
IExternalProcess - Hierarchy List 43
IExternalProcess - Member Functions and Data by Group 43
IExternalProcess - Inherited Member Functions and Data 48
IHandle 48
IHandle - Hierarchy List 48
IHandle - Member Functions and Data by Group 48
IHandle - Type Definitions 49
IHandle - Associated Globals 50
IHandle - Inherited Member Functions and Data 50
IInterest 50
IInterest - Hierarchy List 50
IInterest - Member Functions and Data by Group 50
IInterest - Inherited Member Functions and Data 52
IModuleHandle 53
IModuleHandle - Hierarchy List 53
IModuleHandle - Member Functions and Data by Group 53
IModuleHandle - Type Definitions 54
IModuleHandle - Associated Globals 55
IModuleHandle - Inherited Member Functions and Data 56

© Copyright IBM Corp. 2001

IMRefCounted 56
IMRefCounted - Hierarchy List 56
IMRefCounted - Member Functions and Data by Group 57
IMRefCounted - Inherited Member Functions and Data 58
INonGUIApplication 58
INonGUIApplication - Hierarchy List 58
INonGUIApplication - Member Functions and Data by Group 58
INonGUIApplication - Enumerations 60
INonGUIApplication - Inherited Member Functions and Data 60
INonGUIThread 60
INonGUIThread - Hierarchy List 61
INonGUIThread - Member Functions and Data by Group 61
INonGUIThread - Enumerations 72
INonGUIThread - Type Definitions 72
INonGUIThread - Inherited Member Functions and Data 72
INonGUIThread::Cursor 73
INonGUIThread::Cursor - Hierarchy List . . . 73
INonGUIThread::Cursor - Member Functions and Data by Group 73
INonGUIThread::Cursor - Inherited Member Functions and Data 74
INotificationEvent 74
INotificationEvent - Hierarchy List 74
INotificationEvent - Member Functions and Data by Group 75
INotificationEvent - Inherited Member Functions and Data 76
INotificationEventFor 76
INotificationEventFor - Hierarchy List 76
INotificationEventFor - Member Functions and Data by Group 77
INotificationEventFor - Inherited Member Functions and Data 78
INotifier 78
INotifier - Hierarchy List 79
INotifier - Member Functions and Data by Group 79
INotifier - Inherited Member Functions and Data 82
INotifierAddress 82
INotifierAddress - Hierarchy List 82
INotifierAddress - Member Functions and Data by Group 83
INotifierAddress - Inherited Member Functions and Data 84
IObserver 84
IObserver - Hierarchy List 84
IObserver - Member Functions and Data by Group 84
IObserver - Inherited Member Functions and Data 86
IObserverConnectionTo 86
IObserverConnectionTo - Hierarchy List 86
IObserverConnectionTo - Member Functions and Data by Group 87
IObserverConnectionTo - Type Definitions . . . 87
IObserverConnectionTo - Inherited Member Functions and Data 87
IObserverForConnectionTo 88
IObserverForConnectionTo - Hierarchy List . . 88
IObserverForConnectionTo - Member Functions and Data by Group 88
IObserverForConnectionTo - Type Definitions . . 89
IObserverForConnectionTo - Inherited Member Functions and Data 89
IObserverList 89
IObserverList - Hierarchy List 89
IObserverList - Member Functions and Data by Group 90
IObserverList - Inherited Member Functions and Data 91
ioc 91
ioc - Hierarchy List 91
ioc - Member Functions and Data by Group . . 92
ioc - Enumerations 92
ioc - Type Definitions 92
ioc - Inherited Member Functions and Data. . . 96
ioc::itimeval 96
ioc::itimeval - Hierarchy List. 96
ioc::itimeval - Member Functions and Data by Group 96
ioc::itimeval - Inherited Member Functions and Data 97
ioc::itimezone. 97
ioc::itimezone - Hierarchy List 97
ioc::itimezone - Member Functions and Data by Group 97
ioc::itimezone - Inherited Member Functions and Data 97
ioc::statvfs 97
ioc::statvfs - Hierarchy List 98
ioc::statvfs - Member Functions and Data by Group 98
ioc::statvfs - Inherited Member Functions and Data 99
IPerformanceSettings 99
IPerformanceSettings - Hierarchy List 99
IPerformanceSettings - Member Functions and Data by Group 99
IPerformanceSettings - Enumerations 100
IPerformanceSettings - Inherited Member Functions and Data 101
IPrimalLock 101
IPrimalLock - Hierarchy List 101
IPrimalLock - Member Functions and Data by Group 101
IPrimalLock - Inherited Member Functions and Data 102
IPrivateCondition 102
IPrivateCondition - Hierarchy List 102
IPrivateCondition - Member Functions and Data by Group 102
IPrivateCondition - Inherited Member Functions and Data 104
IPrivateResource 104
IPrivateResource - Hierarchy List. 105

IPrivateResource - Member Functions and Data by Group . . . 105
IPrivateResource - Inherited Member Functions and Data . . . 106
IPrivateSemaphoreHandle . . . 107
IPrivateSemaphoreHandle - Hierarchy List . . . 107
IPrivateSemaphoreHandle - Member Functions and Data by Group . . . 107
IPrivateSemaphoreHandle - Type Definitions 108
IPrivateSemaphoreHandle - Associated Globals 108
IPrivateSemaphoreHandle - Inherited Member Functions and Data . . . 109
IProcessId . . . 109
IProcessId - Hierarchy List . . . 109
IProcessId - Member Functions and Data by Group . . . 109
IProcessId - Type Definitions . . . 110
IProcessId - Associated Globals . . . 110
IProcessId - Inherited Member Functions and Data . . . 111
IResource . . . 111
IResource - Hierarchy List . . . 111
IResource - Member Functions and Data by Group . . . 111
IResource - Inherited Member Functions and Data . . . 113
IResourceId . . . 113
IResourceId - Hierarchy List . . . 113
IResourceId - Member Functions and Data by Group . . . 113
IResourceId - Inherited Member Functions and Data . . . 115
IResourceLibrary . . . 115
IResourceLibrary - Hierarchy List. . . . 115
IResourceLibrary - Member Functions and Data by Group . . . 115
IResourceLibrary - Inherited Member Functions and Data . . . 124
IResourceLock . . . 124
IResourceLock - Hierarchy List . . . 124
IResourceLock - Member Functions and Data by Group . . . 124
IResourceLock - Inherited Member Functions and Data . . . 125
ISharedCondition . . . 126
ISharedCondition - Hierarchy List . . . 126
ISharedCondition - Member Functions and Data by Group . . . 126
ISharedCondition - Inherited Member Functions and Data . . . 129
ISharedResource . . . 129
ISharedResource - Hierarchy List . . . 130
ISharedResource - Member Functions and Data by Group . . . 130
ISharedResource - Inherited Member Functions and Data . . . 132
ISharedSemaphoreHandle . . . 133
ISharedSemaphoreHandle - Hierarchy List . . . 133
ISharedSemaphoreHandle - Member Functions and Data by Group . . . 133
ISharedSemaphoreHandle - Type Definitions 134
ISharedSemaphoreHandle - Associated Globals 134
ISharedSemaphoreHandle - Inherited Member Functions and Data . . . 134
IStandardNotifier . . . 135
IStandardNotifier - Hierarchy List . . . 135
IStandardNotifier - Member Functions and Data by Group . . . 135
IStandardNotifier - Inherited Member Functions and Data . . . 138
IStdDeque . . . 139
IStdDeque - Hierarchy List . . . 139
IStdDeque - Member Functions and Data by Group . . . 139
IStdDeque - Type Definitions . . . 139
IStdDeque - Inherited Member Functions and Data . . . 139
IStdEqualitySequence. . . . 140
IStdEqualitySequence - Hierarchy List . . . 140
IStdEqualitySequence - Member Functions and Data by Group . . . 140
IStdEqualitySequence - Type Definitions . . . 140
IStdEqualitySequence - Inherited Member Functions and Data . . . 140
IStdKeySet . . . 140
IStdKeySet - Hierarchy List. . . . 140
IStdKeySet - Member Functions and Data by Group . . . 140
IStdKeySet - Type Definitions . . . 141
IStdKeySet - Inherited Member Functions and Data . . . 141
IStdKeySortedSet . . . 141
IStdKeySortedSet - Hierarchy List . . . 141
IStdKeySortedSet - Member Functions and Data by Group . . . 141
IStdKeySortedSet - Type Definitions . . . 141
IStdKeySortedSet - Inherited Member Functions and Data . . . 142
IStdKeySortedSetAsTable . . . 142
IStdKeySortedSetAsTable - Hierarchy List . . . 142
IStdKeySortedSetAsTable - Member Functions and Data by Group . . . 142
IStdKeySortedSetAsTable - Type Definitions . . . 142
IStdKeySortedSetAsTable - Inherited Member Functions and Data . . . 142
IStdKeySortedSetOnSortedTabularSequence . . . 143
IStdKeySortedSetOnSortedTabularSequence - Hierarchy List . . . 143
IStdKeySortedSetOnSortedTabularSequence - Member Functions and Data by Group . . . 143
IStdKeySortedSetOnSortedTabularSequence - Type Definitions . . . 143
IStdKeySortedSetOnSortedTabularSequence - Inherited Member Functions and Data . . . 143
IStdQueue . . . 143
IStdQueue - Hierarchy List . . . 143
IStdQueue - Member Functions and Data by Group . . . 144
IStdQueue - Type Definitions . . . 144
IStdQueue - Inherited Member Functions and Data . . . 144
IStdSequence . . . 144

IStdSequence - Hierarchy List 144
IStdSequence - Member Functions and Data by Group 144
IStdSequence - Type Definitions 144
IStdSequence - Inherited Member Functions and Data 145
IStdSet 145
IStdSet - Hierarchy List 145
IStdSet - Member Functions and Data by Group 145
IStdSet - Type Definitions 145
IStdSet - Inherited Member Functions and Data 145
IStdSortedSet 146
IStdSortedSet - Hierarchy List 146
IStdSortedSet - Member Functions and Data by Group 146
IStdSortedSet - Type Definitions 146
IStdSortedSet - Inherited Member Functions and Data 146
IStdStack 146
IStdStack - Hierarchy List 146
IStdStack - Member Functions and Data by Group 146
IStdStack - Type Definitions 147
IStdStack - Inherited Member Functions and Data 147
ISynchronized 147
ISynchronized - Hierarchy List 148
ISynchronized - Member Functions and Data by Group 148
ISynchronized - Inherited Member Functions and Data 149
IThreadFn 149
IThreadFn - Hierarchy List 150
IThreadFn - Member Functions and Data by Group 150
IThreadFn - Inherited Member Functions and Data 150
IThreadHandle 151
IThreadHandle - Hierarchy List 151
IThreadHandle - Member Functions and Data by Group 151
IThreadHandle - Type Definitions 152
IThreadHandle - Associated Globals 152
IThreadHandle - Inherited Member Functions and Data 153
IThreadId 153
IThreadId - Hierarchy List 153
IThreadId - Member Functions and Data by Group 153
IThreadId - Associated Globals 155
IThreadId - Inherited Member Functions and Data 156
IThreadLocalBase 156
IThreadLocalBase - Hierarchy List 156
IThreadLocalBase - Member Functions and Data by Group 156
IThreadLocalBase - Inherited Member Functions and Data 158
IThreadLocalStorage 158
IThreadLocalStorage - Hierarchy List 159
IThreadLocalStorage - Member Functions and Data by Group 159
IThreadLocalStorage - Inherited Member Functions and Data 161
IThreadMemberFn. 162
IThreadMemberFn - Hierarchy List 162
IThreadMemberFn - Member Functions and Data by Group 163
IThreadMemberFn - Inherited Member Functions and Data 163
ITimer. 164
ITimer - Hierarchy List 164
ITimer - Member Functions and Data by Group 164
ITimer - Inherited Member Functions and Data 168
ITimer::Cursor 169
ITimer::Cursor - Hierarchy List 169
ITimer::Cursor - Member Functions and Data by Group 169
ITimer::Cursor - Inherited Member Functions and Data 170
ITimerFn 170
ITimerFn - Hierarchy List 170
ITimerFn - Member Functions and Data by Group 170
ITimerFn - Inherited Member Functions and Data 171
ITimerMemberFn 172
ITimerMemberFn - Hierarchy List 172
ITimerMemberFn - Member Functions and Data by Group. 172
ITimerMemberFn - Inherited Member Functions and Data 173
ITimerMemberFn0. 173
ITimerMemberFn0 - Hierarchy List 174
ITimerMemberFn0 - Member Functions and Data by Group 174
ITimerMemberFn0 - Inherited Member Functions and Data 175
IVersion 175
IVersion - Hierarchy List 175
IVersion - Member Functions and Data by Group 176
IVersion - Inherited Member Functions and Data 179
IWaitAbandoned 179
IWaitAbandoned - Hierarchy List. 179
IWaitAbandoned - Member Functions and Data by Group. 179
IWaitAbandoned - Inherited Member Functions and Data 180
std 181
std - Hierarchy List 181
std - Member Functions and Data by Group . . 181
std - Inherited Member Functions and Data . . 181

Chapter 2. Streams 183

IAddressAlreadyInContext 183
IAddressAlreadyInContext - Hierarchy List . . 183
IAddressAlreadyInContext - Member Functions and Data by Group 183
IAddressAlreadyInContext - Inherited Member Functions and Data 184

IDataStream 185
IDataStream - Hierarchy List 185
IDataStream - Member Functions and Data by Group 185
IDataStream - Enumerations 193
IDataStream - Inherited Member Functions and Data 193
IEndOfStream 194
IEndOfStream - Hierarchy List 194
IEndOfStream - Member Functions and Data by Group 194
IEndOfStream - Inherited Member Functions and Data 195
IInvalidContextNumber 195
IInvalidContextNumber - Hierarchy List . . . 196
IInvalidContextNumber - Member Functions and Data by Group 196
IInvalidContextNumber - Inherited Member Functions and Data 197
IInvalidDataOnStream 197
IInvalidDataOnStream - Hierarchy List 197
IInvalidDataOnStream - Member Functions and Data by Group 198
IInvalidDataOnStream - Inherited Member Functions and Data 198
IMStreamable 199
IMStreamable - Hierarchy List. 200
IMStreamable - Member Functions and Data by Group 200
IMStreamable - Inherited Member Functions and Data 202
IStreamContextFrame. 202
IStreamContextFrame - Hierarchy List 203
IStreamContextFrame - Member Functions and Data by Group 203
IStreamContextFrame - Inherited Member Functions and Data 204
IStreamInFrame 204
IStreamInFrame - Hierarchy List 205
IStreamInFrame - Member Functions and Data by Group. 205
IStreamInFrame - Inherited Member Functions and Data 206
IStreamIO 206
IStreamIO - Hierarchy List 206
IStreamIO - Member Functions and Data by Group 206
IStreamIO - Inherited Member Functions and Data 215
IStreamModule. 215
IStreamModule - Hierarchy List 216
IStreamModule - Member Functions and Data by Group. 216
IStreamModule - Inherited Member Functions and Data 216
IStreamOutFrame 216
IStreamOutFrame - Hierarchy List 217
IStreamOutFrame - Member Functions and Data by Group. 217
IStreamOutFrame - Inherited Member Functions and Data 218
IUnknownTypeOnStream 218
IUnknownTypeOnStream - Hierarchy List . . . 218
IUnknownTypeOnStream - Member Functions and Data by Group 218
IUnknownTypeOnStream - Inherited Member Functions and Data 219

Chapter 3. File Systems 221

ICompoundName 221
ICompoundName - Hierarchy List 221
ICompoundName - Member Functions and Data by Group. 221
ICompoundName - Inherited Member Functions and Data 225
ICompoundNameParser. 225
ICompoundNameParser - Hierarchy List . . . 225
ICompoundNameParser - Member Functions and Data by Group 226
ICompoundNameParser - Type Definitions . . 227
ICompoundNameParser - Inherited Member Functions and Data 227
IDirectory 228
IDirectory - Hierarchy List 228
IDirectory - Member Functions and Data by Group 228
IDirectory - Inherited Member Functions and Data 233
IDirectoryIterator 234
IDirectoryIterator - Hierarchy List 234
IDirectoryIterator - Member Functions and Data by Group. 234
IDirectoryIterator - Inherited Member Functions and Data 237
IEntityInUse. 237
IEntityInUse - Hierarchy List 237
IEntityInUse - Member Functions and Data by Group 237
IEntityInUse - Inherited Member Functions and Data 238
IEntityInvalid 239
IEntityInvalid - Hierarchy List. 239
IEntityInvalid - Member Functions and Data by Group 239
IEntityInvalid - Inherited Member Functions and Data 240
IEntityTypeMismatch. 241
IEntityTypeMismatch - Hierarchy List 241
IEntityTypeMismatch - Member Functions and Data by Group 241
IEntityTypeMismatch - Inherited Member Functions and Data 243
IFifoFile 243
IFifoFile - Hierarchy List 244
IFifoFile - Member Functions and Data by Group 244
IFifoFile - Enumerations. 247
IFifoFile - Inherited Member Functions and Data 248
IFifoStdIOStream 249
IFifoStdIOStream - Hierarchy List 249
IFifoStdIOStream - Member Functions and Data by Group. 249

IFifoStdIOStream - Inherited Member Functions and Data 254
IFifoUslIOStream 254
IFifoUslIOStream - Hierarchy List 254
IFifoUslIOStream - Member Functions and Data by Group 254
IFifoUslIOStream - Inherited Member Functions and Data 259
IFile 261
IFile - Hierarchy List 261
IFile - Member Functions and Data by Group 261
IFile - Enumerations 265
IFile - Inherited Member Functions and Data 266
IFileOperation 267
IFileOperation - Hierarchy List 267
IFileOperation - Member Functions and Data by Group 267
IFileOperation - Enumerations. 271
IFileOperation - Inherited Member Functions and Data 271
IFileOperationCancelled 271
IFileOperationCancelled - Hierarchy List . . . 271
IFileOperationCancelled - Member Functions and Data by Group 272
IFileOperationCancelled - Inherited Member Functions and Data 273
IFileStdIOStream 273
IFileStdIOStream - Hierarchy List. 274
IFileStdIOStream - Member Functions and Data by Group. 274
IFileStdIOStream - Inherited Member Functions and Data 278
IFileSystemCopier 279
IFileSystemCopier - Hierarchy List 279
IFileSystemCopier - Member Functions and Data by Group 279
IFileSystemCopier - Inherited Member Functions and Data 280
IFileSystemEntity 281
IFileSystemEntity - Hierarchy List 281
IFileSystemEntity - Member Functions and Data by Group. 281
IFileSystemEntity - Enumerations. 287
IFileSystemEntity - Inherited Member Functions and Data 287
IFileSystemMover 287
IFileSystemMover - Hierarchy List 288
IFileSystemMover - Member Functions and Data by Group. 288
IFileSystemMover - Inherited Member Functions and Data 289
IFileUslIOStream 289
IFileUslIOStream - Hierarchy List 290
IFileUslIOStream - Member Functions and Data by Group. 290
IFileUslIOStream - Inherited Member Functions and Data 294
INativeObject 296
INativeObject - Hierarchy List. 297
INativeObject - Member Functions and Data by Group 297
INativeObject - Inherited Member Functions and Data 299
IOS400PathNameParser 300
IOS400PathNameParser - Hierarchy List . . . 300
IOS400PathNameParser - Member Functions and Data by Group 301
IOS400PathNameParser - Inherited Member Functions and Data 304
IPathName 305
IPathName - Hierarchy List 305
IPathName - Member Functions and Data by Group 305
IPathName - Inherited Member Functions and Data 308
IRootDirectoryIterator 308
IRootDirectoryIterator - Hierarchy List 309
IRootDirectoryIterator - Member Functions and Data by Group 309
IRootDirectoryIterator - Inherited Member Functions and Data 311
ITimeStamp 311
ITimeStamp - Hierarchy List 311
ITimeStamp - Member Functions and Data by Group 311
ITimeStamp - Associated Globals 315
ITimeStamp - Inherited Member Functions and Data 316
ITypeRepresentation 316
ITypeRepresentation - Hierarchy List 316
ITypeRepresentation - Member Functions and Data by Group 316
ITypeRepresentation - Inherited Member Functions and Data 316
IUnixPathNameParser 317
IUnixPathNameParser - Hierarchy List 317
IUnixPathNameParser - Member Functions and Data by Group 317
IUnixPathNameParser - Inherited Member Functions and Data 318
IVolume 319
IVolume - Hierarchy List 319
IVolume - Member Functions and Data by Group 319
IVolume - Enumerations. 322
IVolume - Inherited Member Functions and Data 322
IVolumeIterator. 323
IVolumeIterator - Hierarchy List 323
IVolumeIterator - Member Functions and Data by Group. 323
IVolumeIterator - Inherited Member Functions and Data 325
IVolumeOffline 326
IVolumeOffline - Hierarchy List 326
IVolumeOffline - Member Functions and Data by Group. 326
IVolumeOffline - Inherited Member Functions and Data 327
IWin32PathNameParser 328
IWin32PathNameParser - Hierarchy List . . . 328

IWin32PathNameParser - Member Functions and Data by Group 328
IWin32PathNameParser - Inherited Member Functions and Data 329

Chapter 4. Text and International Classes 331

codecvt_base 331
codecvt_base - Hierarchy List 331
codecvt_base - Member Functions and Data by Group 331
codecvt_base - Enumerations 331
codecvt_base - Inherited Member Functions and Data 331
I0String 331
I0String - Hierarchy List 332
I0String - Member Functions and Data by Group 332
I0String - Associated Globals 346
I0String - Inherited Member Functions and Data 347
IBinaryFormat 349
IBinaryFormat - Hierarchy List 350
IBinaryFormat - Member Functions and Data by Group 350
IBinaryFormat - Associated Globals 351
IBinaryFormat - Inherited Member Functions and Data 351
IBitwiseCollation 351
IBitwiseCollation - Hierarchy List 351
IBitwiseCollation - Member Functions and Data by Group 351
IBitwiseCollation - Inherited Member Functions and Data 353
IBuffer. 354
IBuffer - Hierarchy List 354
IBuffer - Member Functions and Data by Group 354
IBuffer - Type Definitions 368
IBuffer - Inherited Member Functions and Data 368
ICCSId 369
ICCSId - Hierarchy List 369
ICCSId - Member Functions and Data by Group 369
ICCSId - Inherited Member Functions and Data 371
ICharacterPropertyIterator 371
ICharacterPropertyIterator - Hierarchy List . . 371
ICharacterPropertyIterator - Member Functions and Data by Group 371
ICharacterPropertyIterator - Inherited Member Functions and Data 373
ICharacterReference 374
ICharacterReference - Hierarchy List 374
ICharacterReference - Member Functions and Data by Group 374
ICharacterReference - Inherited Member Functions and Data 376
ICharacterSetIterator 376
ICharacterSetIterator - Hierarchy List 376
ICharacterSetIterator - Member Functions and Data by Group 376
ICharacterSetIterator - Inherited Member Functions and Data 379
ICharacterStyle 379
ICharacterStyle - Hierarchy List 379
ICharacterStyle - Member Functions and Data by Group. 379
ICharacterStyle - Inherited Member Functions and Data 380
ICnrDate 381
ICnrDate - Hierarchy List 381
ICnrDate - Member Functions and Data by Group 381
ICnrDate - Inherited Member Functions and Data 381
ICnrTime 382
ICnrTime - Hierarchy List 382
ICnrTime - Member Functions and Data by Group 382
ICnrTime - Inherited Member Functions and Data 382
ICollation 383
ICollation - Hierarchy List 384
ICollation - Member Functions and Data by Group 384
ICollation - Enumerations 390
ICollation - Inherited Member Functions and Data 390
ICollationIterator 391
ICollationIterator - Hierarchy List 391
ICollationIterator - Member Functions and Data by Group. 391
ICollationIterator - Enumerations. 394
ICollationIterator - Inherited Member Functions and Data 394
IConstTextIterator 394
IConstTextIterator - Hierarchy List 394
IConstTextIterator - Member Functions and Data by Group. 395
IConstTextIterator - Inherited Member Functions and Data 399
IDate 400
IDate - Hierarchy List 400
IDate - Member Functions and Data by Group 400
IDate - Type Definitions 410
IDate - Associated Globals 410
IDate - Inherited Member Functions and Data 410
IDBCSBuffer. 411
IDBCSBuffer - Hierarchy List 411
IDBCSBuffer - Member Functions and Data by Group 411
IDBCSBuffer - Inherited Member Functions and Data 423
IDecimalFormat 425
IDecimalFormat - Hierarchy List 426
IDecimalFormat - Member Functions and Data by Group. 426
IDecimalFormat - Associated Globals 430
IDecimalFormat - Inherited Member Functions and Data 430
IFastTextIterator 430
IFastTextIterator - Hierarchy List 431
IFastTextIterator - Member Functions and Data by Group. 431

IFastTextIterator - Inherited Member Functions and Data 435
IHexadecimalFormat 435
IHexadecimalFormat - Hierarchy List 436
IHexadecimalFormat - Member Functions and Data by Group 437
IHexadecimalFormat - Inherited Member Functions and Data 438
ILineBreakConverter 438
ILineBreakConverter - Hierarchy List 439
ILineBreakConverter - Member Functions and Data by Group 439
ILineBreakConverter - Enumerations 440
ILineBreakConverter - Inherited Member Functions and Data 440
ILocaleKey 440
ILocaleKey - Hierarchy List. 441
ILocaleKey - Member Functions and Data by Group 441
ILocaleKey - Enumerations 447
ILocaleKey - Inherited Member Functions and Data 447
ILocaleKeyIterator 448
ILocaleKeyIterator - Hierarchy List 448
ILocaleKeyIterator - Member Functions and Data by Group 448
ILocaleKeyIterator - Inherited Member Functions and Data 451
IMTabRuler 451
IMTabRuler - Hierarchy List 451
IMTabRuler - Member Functions and Data by Group 451
IMTabRuler - Inherited Member Functions and Data 452
IParagraphEndIndentStyle 453
IParagraphEndIndentStyle - Hierarchy List . . 453
IParagraphEndIndentStyle - Member Functions and Data by Group 453
IParagraphEndIndentStyle - Inherited Member Functions and Data 455
IParagraphFirstIndentStyle 456
IParagraphFirstIndentStyle - Hierarchy List . . 456
IParagraphFirstIndentStyle - Member Functions and Data by Group 456
IParagraphFirstIndentStyle - Inherited Member Functions and Data 458
IParagraphJustificationStyle 459
IParagraphJustificationStyle - Hierarchy List . . 459
IParagraphJustificationStyle - Member Functions and Data by Group 459
IParagraphJustificationStyle - Enumerations . . 461
IParagraphJustificationStyle - Inherited Member Functions and Data 461
IParagraphSpreadStyle 462
IParagraphSpreadStyle - Hierarchy List. 462
IParagraphSpreadStyle - Member Functions and Data by Group 462
IParagraphSpreadStyle - Enumerations 464
IParagraphSpreadStyle - Inherited Member Functions and Data 464
IParagraphStartIndentStyle 465
IParagraphStartIndentStyle - Hierarchy List . . 465
IParagraphStartIndentStyle - Member Functions and Data by Group 465
IParagraphStartIndentStyle - Inherited Member Functions and Data 467
IParagraphStyle 468
IParagraphStyle - Hierarchy List 468
IParagraphStyle - Member Functions and Data by Group. 469
IParagraphStyle - Inherited Member Functions and Data 470
IStandardTabRuler. 470
IStandardTabRuler - Hierarchy List 470
IStandardTabRuler - Member Functions and Data by Group 470
IStandardTabRuler - Inherited Member Functions and Data 473
IString. 473
IString - Hierarchy List 475
IString - Member Functions and Data by Group 475
IString - Type Definitions 517
IString - Associated Globals 522
IString - Inherited Member Functions and Data 531
IStringEnum. 532
IStringEnum - Hierarchy List 532
IStringEnum - Member Functions and Data by Group 532
IStringEnum - Type Definitions 532
IStringEnum - Inherited Member Functions and Data 533
IStringGenerator 533
IStringGenerator - Hierarchy List. 534
IStringGenerator - Member Functions and Data by Group. 534
IStringGenerator - Inherited Member Functions and Data 535
IStringGeneratorasString. 536
IStringGeneratorasString - Hierarchy List . . . 536
IStringGeneratorasString - Member Functions and Data by Group 536
IStringGeneratorasString - Inherited Member Functions and Data 537
IStringGeneratorFn 537
IStringGeneratorFn - Hierarchy List 538
IStringGeneratorFn - Member Functions and Data by Group 538
IStringGeneratorFn - Inherited Member Functions and Data 538
IStringGeneratorMemberFn. 539
IStringGeneratorMemberFn - Hierarchy List . . 539
IStringGeneratorMemberFn - Member Functions and Data by Group 540
IStringGeneratorMemberFn - Inherited Member Functions and Data 540
IStringGeneratorRefMemberFn 541
IStringGeneratorRefMemberFn - Hierarchy List 542
IStringGeneratorRefMemberFn - Member Functions and Data by Group 542
IStringGeneratorRefMemberFn - Inherited Member Functions and Data 543
IStringParser 543

IStringParser - Hierarchy List 544
IStringParser - Member Functions and Data by Group 545
IStringParser - Enumerations 548
IStringParser - Associated Globals 548
IStringParser - Inherited Member Functions and Data 549
IStringParser::SkipWords 550
IStringParser::SkipWords - Hierarchy List . . . 550
IStringParser::SkipWords - Member Functions and Data by Group 550
IStringParser::SkipWords - Inherited Member Functions and Data 550
IStringTest 551
IStringTest - Hierarchy List 551
IStringTest - Member Functions and Data by Group 551
IStringTest - Enumerations 552
IStringTest - Type Definitions 553
IStringTest - Inherited Member Functions and Data 553
IStringTestMemberFn 553
IStringTestMemberFn - Hierarchy List 553
IStringTestMemberFn - Member Functions and Data by Group 553
IStringTestMemberFn - Type Definitions . . . 554
IStringTestMemberFn - Inherited Member Functions and Data 555
ITabRulerStyle 555
ITabRulerStyle - Hierarchy List 555
ITabRulerStyle - Member Functions and Data by Group 555
ITabRulerStyle - Inherited Member Functions and Data 557
ITabStop 557
ITabStop - Hierarchy List 558
ITabStop - Member Functions and Data by Group 558
ITabStop - Enumerations 559
ITabStop - Inherited Member Functions and Data 560
IText 560
IText - Hierarchy List 560
IText - Member Functions and Data by Group 560
IText - Type Definitions 594
IText - Inherited Member Functions and Data 594
ITextBitmapStyle 594
ITextBitmapStyle - Hierarchy List 595
ITextBitmapStyle - Member Functions and Data by Group 595
ITextBitmapStyle - Inherited Member Functions and Data 597
ITextBoldfaceStyle 598
ITextBoldfaceStyle - Hierarchy List 598
ITextBoldfaceStyle - Member Functions and Data by Group 598
ITextBoldfaceStyle - Inherited Member Functions and Data 600
ITextBoundary 600
ITextBoundary - Hierarchy List 601
ITextBoundary - Member Functions and Data by Group 602
ITextBoundary - Type Definitions 605
ITextBoundary - Inherited Member Functions and Data 606
ITextColorStyle 606
ITextColorStyle - Hierarchy List 606
ITextColorStyle - Member Functions and Data by Group 606
ITextColorStyle - Inherited Member Functions and Data 609
ITextFixedPitchStyle 610
ITextFixedPitchStyle - Hierarchy List 610
ITextFixedPitchStyle - Member Functions and Data by Group 610
ITextFixedPitchStyle - Inherited Member Functions and Data 612
ITextItalicStyle 613
ITextItalicStyle - Hierarchy List 613
ITextItalicStyle - Member Functions and Data by Group 613
ITextItalicStyle - Inherited Member Functions and Data 614
ITextIterator 615
ITextIterator - Hierarchy List 615
ITextIterator - Member Functions and Data by Group 616
ITextIterator - Inherited Member Functions and Data 620
ITextOutlineStyle 620
ITextOutlineStyle - Hierarchy List 620
ITextOutlineStyle - Member Functions and Data by Group 621
ITextOutlineStyle - Inherited Member Functions and Data 622
ITextPlacement 623
ITextPlacement - Hierarchy List 623
ITextPlacement - Member Functions and Data by Group 623
ITextPlacement - Enumerations 626
ITextPlacement - Inherited Member Functions and Data 626
ITextPointSizeStyle 626
ITextPointSizeStyle - Hierarchy List 626
ITextPointSizeStyle - Member Functions and Data by Group 626
ITextPointSizeStyle - Inherited Member Functions and Data 628
ITextStrikethroughStyle 629
ITextStrikethroughStyle - Hierarchy List . . . 629
ITextStrikethroughStyle - Member Functions and Data by Group 629
ITextStrikethroughStyle - Inherited Member Functions and Data 631
ITextStyle 632
ITextStyle - Hierarchy List 632
ITextStyle - Member Functions and Data by Group 632
ITextStyle - Enumerations 634
ITextStyle - Inherited Member Functions and Data 634

ITextStyleRunIterator 634
ITextStyleRunIterator - Hierarchy List 635
ITextStyleRunIterator - Member Functions and Data by Group 635
ITextStyleRunIterator - Type Definitions . . . 637
ITextStyleRunIterator - Inherited Member Functions and Data 638
ITextStyleSet. 638
ITextStyleSet - Hierarchy List 638
ITextStyleSet - Member Functions and Data by Group 638
ITextStyleSet - Inherited Member Functions and Data 645
ITextSuperSubscriptStyle 646
ITextSuperSubscriptStyle - Hierarchy List . . . 646
ITextSuperSubscriptStyle - Member Functions and Data by Group 646
ITextSuperSubscriptStyle - Inherited Member Functions and Data 648
ITextTypefaceStyle. 649
ITextTypefaceStyle - Hierarchy List 649
ITextTypefaceStyle - Member Functions and Data by Group 649
ITextTypefaceStyle - Inherited Member Functions and Data 651
ITextUnderlineStyle 652
ITextUnderlineStyle - Hierarchy List. 652
ITextUnderlineStyle - Member Functions and Data by Group 652
ITextUnderlineStyle - Inherited Member Functions and Data 654
ITextUneditableStyle 655
ITextUneditableStyle - Hierarchy List 655
ITextUneditableStyle - Member Functions and Data by Group 655
ITextUneditableStyle - Inherited Member Functions and Data 656
ITime 657
ITime - Hierarchy List 658
ITime - Member Functions and Data by Group 658
ITime - Associated Globals 663
ITime - Inherited Member Functions and Data 663
ITranscoder 664
ITranscoder - Hierarchy List 666
ITranscoder - Member Functions and Data by Group 666
ITranscoder - Enumerations 674
ITranscoder - Type Definitions. 675
ITranscoder - Inherited Member Functions and Data 675
IUnicode 675
IUnicode - Hierarchy List 675
IUnicode - Member Functions and Data by Group 676
IUnicode - Enumerations 680
IUnicode - Inherited Member Functions and Data 681
UAPLFunctionalSymbol 681
UAPLFunctionalSymbol - Hierarchy List . . . 681
UAPLFunctionalSymbol - Member Functions and Data by Group 681
UAPLFunctionalSymbol - Enumerations . . . 681
UAPLFunctionalSymbol - Inherited Member Functions and Data 681
UArabic 682
UArabic - Hierarchy List 682
UArabic - Member Functions and Data by Group 682
UArabic - Enumerations. 682
UArabic - Inherited Member Functions and Data 682
UArabicCompatibility 682
UArabicCompatibility - Hierarchy List 682
UArabicCompatibility - Member Functions and Data by Group 682
UArabicCompatibility - Enumerations 682
UArabicCompatibility - Inherited Member Functions and Data 682
UArabicLigature 683
UArabicLigature - Hierarchy List. 683
UArabicLigature - Member Functions and Data by Group. 683
UArabicLigature - Enumerations 683
UArabicLigature - Inherited Member Functions and Data 683
UArmenian 683
UArmenian - Hierarchy List 683
UArmenian - Member Functions and Data by Group 683
UArmenian - Enumerations 683
UArmenian - Inherited Member Functions and Data 683
UArmenianSmallLigature 684
UArmenianSmallLigature - Hierarchy List. . . 684
UArmenianSmallLigature - Member Functions and Data by Group 684
UArmenianSmallLigature - Enumerations . . . 684
UArmenianSmallLigature - Inherited Member Functions and Data 684
UArrow 684
UArrow - Hierarchy List 684
UArrow - Member Functions and Data by Group 684
UArrow - Enumerations. 684
UArrow - Inherited Member Functions and Data 685
UASCII 685
UASCII - Hierarchy List. 685
UASCII - Member Functions and Data by Group 685
UASCII - Enumerations 685
UASCII - Inherited Member Functions and Data 685
UBengali 685
UBengali - Hierarchy List 685
UBengali - Member Functions and Data by Group 685
UBengali - Enumerations 685
UBengali - Inherited Member Functions and Data 686
UBlocks 686
UBlocks - Hierarchy List 686
UBlocks - Member Functions and Data by Group 686

UBlocks - Enumerations 686
UBlocks - Inherited Member Functions and Data 686
UBopomofo 686
UBopomofo - Hierarchy List 686
UBopomofo - Member Functions and Data by Group 686
UBopomofo - Enumerations 686
UBopomofo - Inherited Member Functions and Data 687
UBoxDrawings 687
UBoxDrawings - Hierarchy List 687
UBoxDrawings - Member Functions and Data by Group 687
UBoxDrawings - Enumerations 687
UBoxDrawings - Inherited Member Functions and Data 687
UCircledDigit 687
UCircledDigit - Hierarchy List 687
UCircledDigit - Member Functions and Data by Group 688
UCircledDigit - Enumerations 688
UCircledDigit - Inherited Member Functions and Data 688
UCircledHangul 688
UCircledHangul - Hierarchy List 688
UCircledHangul - Member Functions and Data by Group 688
UCircledHangul - Enumerations 688
UCircledHangul - Inherited Member Functions and Data 688
UCircledIdeograph 688
UCircledIdeograph - Hierarchy List 689
UCircledIdeograph - Member Functions and Data by Group 689
UCircledIdeograph - Enumerations 689
UCircledIdeograph - Inherited Member Functions and Data 689
UCircledKatakana 689
UCircledKatakana - Hierarchy List 689
UCircledKatakana - Member Functions and Data by Group 689
UCircledKatakana - Enumerations 689
UCircledKatakana - Inherited Member Functions and Data 689
UCircledLatin 690
UCircledLatin - Hierarchy List. 690
UCircledLatin - Member Functions and Data by Group 690
UCircledLatin - Enumerations 690
UCircledLatin - Inherited Member Functions and Data 690
UCircledNumber 690
UCircledNumber - Hierarchy List 690
UCircledNumber - Member Functions and Data by Group 690
UCircledNumber - Enumerations 690
UCircledNumber - Inherited Member Functions and Data 690
UCJKCompatibility 691
UCJKCompatibility - Hierarchy List 691
UCJKCompatibility - Member Functions and Data by Group 691
UCJKCompatibility - Enumerations 691
UCJKCompatibility - Inherited Member Functions and Data 691
UCJKRadical 691
UCJKRadical - Hierarchy List 691
UCJKRadical - Member Functions and Data by Group 691
UCJKRadical - Enumerations 691
UCJKRadical - Inherited Member Functions and Data 691
UCJKSquaredAbbreviations 692
UCJKSquaredAbbreviations - Hierarchy List . . 692
UCJKSquaredAbbreviations - Member Functions and Data by Group 692
UCJKSquaredAbbreviations - Enumerations . . 692
UCJKSquaredAbbreviations - Inherited Member Functions and Data 692
UCJKSquaredWords 692
UCJKSquaredWords - Hierarchy List 692
UCJKSquaredWords - Member Functions and Data by Group 692
UCJKSquaredWords - Enumerations. 692
UCJKSquaredWords - Inherited Member Functions and Data 692
UCJKSymbols 693
UCJKSymbols - Hierarchy List 693
UCJKSymbols - Member Functions and Data by Group 693
UCJKSymbols - Enumerations. 693
UCJKSymbols - Inherited Member Functions and Data 693
UCNSCompatibility 693
UCNSCompatibility - Hierarchy List 693
UCNSCompatibility - Member Functions and Data by Group 693
UCNSCompatibility - Enumerations. 693
UCNSCompatibility - Inherited Member Functions and Data 693
UCombining 694
UCombining - Hierarchy List 694
UCombining - Member Functions and Data by Group 694
UCombining - Enumerations 694
UCombining - Inherited Member Functions and Data 694
UControlCode 694
UControlCode - Hierarchy List 694
UControlCode - Member Functions and Data by Group 694
UControlCode - Enumerations. 694
UControlCode - Inherited Member Functions and Data 694
UControlCodePicture. 695
UControlCodePicture - Hierarchy List 695
UControlCodePicture - Member Functions and Data by Group 695
UControlCodePicture - Enumerations 695
UControlCodePicture - Inherited Member Functions and Data 695

UCoptic 695
UCoptic - Hierarchy List 695
UCoptic - Member Functions and Data by Group 695
UCoptic - Enumerations 695
UCoptic - Inherited Member Functions and Data 696
UCurrency 696
UCurrency - Hierarchy List. 696
UCurrency - Member Functions and Data by Group 696
UCurrency - Enumerations 696
UCurrency - Inherited Member Functions and Data 696
UCyrillic 696
UCyrillic - Hierarchy List 696
UCyrillic - Member Functions and Data by Group 697
UCyrillic - Enumerations 697
UCyrillic - Inherited Member Functions and Data 697
UDevanagari 697
UDevanagari - Hierarchy List 697
UDevanagari - Member Functions and Data by Group 697
UDevanagari - Enumerations 697
UDevanagari - Inherited Member Functions and Data 697
UDingbats 697
UDingbats - Hierarchy List 698
UDingbats - Member Functions and Data by Group 698
UDingbats - Enumerations 698
UDingbats - Inherited Member Functions and Data 698
UFullStopDigit 698
UFullStopDigit - Hierarchy List 698
UFullStopDigit - Member Functions and Data by Group. 698
UFullStopDigit - Enumerations 698
UFullStopDigit - Inherited Member Functions and Data 698
UFullStopNumber. 698
UFullStopNumber - Hierarchy List 699
UFullStopNumber - Member Functions and Data by Group 699
UFullStopNumber - Enumerations 699
UFullStopNumber - Inherited Member Functions and Data 699
UFullwidth 699
UFullwidth - Hierarchy List 699
UFullwidth - Member Functions and Data by Group 699
UFullwidth - Enumerations. 699
UFullwidth - Inherited Member Functions and Data 699
UGeneralPunctuation. 700
UGeneralPunctuation - Hierarchy List 700
UGeneralPunctuation - Member Functions and Data by Group 700
UGeneralPunctuation - Enumerations 700
UGeneralPunctuation - Inherited Member Functions and Data 700
UGeometricShapes 700
UGeometricShapes - Hierarchy List 700
UGeometricShapes - Member Functions and Data by Group 700
UGeometricShapes - Enumerations 700
UGeometricShapes - Inherited Member Functions and Data 700
UGeorgian 701
UGeorgian - Hierarchy List. 701
UGeorgian - Member Functions and Data by Group 701
UGeorgian - Enumerations 701
UGeorgian - Inherited Member Functions and Data 701
UGreek 701
UGreek - Hierarchy List 701
UGreek - Member Functions and Data by Group 701
UGreek - Enumerations 701
UGreek - Inherited Member Functions and Data 701
UGujarati. 702
UGujarati - Hierarchy List 702
UGujarati - Member Functions and Data by Group 702
UGujarati - Enumerations 702
UGujarati - Inherited Member Functions and Data 702
UGurmukhi 702
UGurmukhi - Hierarchy List 702
UGurmukhi - Member Functions and Data by Group 702
UGurmukhi - Enumerations 702
UGurmukhi - Inherited Member Functions and Data 702
UHalfwidth 703
UHalfwidth - Hierarchy List 703
UHalfwidth - Member Functions and Data by Group 703
UHalfwidth - Enumerations 703
UHalfwidth - Inherited Member Functions and Data 703
UHalfwidthHangulLetter 703
UHalfwidthHangulLetter - Hierarchy List . . . 703
UHalfwidthHangulLetter - Member Functions and Data by Group 703
UHalfwidthHangulLetter - Enumerations . . . 703
UHalfwidthHangulLetter - Inherited Member Functions and Data 703
UHangul 704
UHangul - Hierarchy List 704
UHangul - Member Functions and Data by Group 704
UHangul - Enumerations 704
UHangul - Inherited Member Functions and Data 704
UHangulChoseong 704
UHangulChoseong - Hierarchy List 704
UHangulChoseong - Member Functions and Data by Group 704
UHangulChoseong - Enumerations 704

UHangulChoseong - Inherited Member Functions and Data 704
UHangulJongseong 705
UHangulJongseong - Hierarchy List 705
UHangulJongseong - Member Functions and Data by Group 705
UHangulJongseong - Enumerations 705
UHangulJongseong - Inherited Member Functions and Data 705
UHangulJungseong 705
UHangulJungseong - Hierarchy List 705
UHangulJungseong - Member Functions and Data by Group 705
UHangulJungseong - Enumerations 705
UHangulJungseong - Inherited Member Functions and Data 705
UHangulLetter 706
UHangulLetter - Hierarchy List 706
UHangulLetter - Member Functions and Data by Group 706
UHangulLetter - Enumerations 706
UHangulLetter - Inherited Member Functions and Data 706
UHangulSyllable 706
UHangulSyllable - Hierarchy List. 706
UHangulSyllable - Member Functions and Data by Group. 706
UHangulSyllable - Enumerations 706
UHangulSyllable - Inherited Member Functions and Data 707
UHangzhouNumeral 707
UHangzhouNumeral - Hierarchy List 707
UHangzhouNumeral - Member Functions and Data by Group 707
UHangzhouNumeral - Enumerations 707
UHangzhouNumeral - Inherited Member Functions and Data 707
UHanNumeral 707
UHanNumeral - Hierarchy List 707
UHanNumeral - Member Functions and Data by Group 708
UHanNumeral - Enumerations 708
UHanNumeral - Inherited Member Functions and Data 708
UHarpoon 708
UHarpoon - Hierarchy List 708
UHarpoon - Member Functions and Data by Group 708
UHarpoon - Enumerations 708
UHarpoon - Inherited Member Functions and Data 708
UHebrew. 708
UHebrew - Hierarchy List 709
UHebrew - Member Functions and Data by Group 709
UHebrew - Enumerations 709
UHebrew - Inherited Member Functions and Data 709
UHiragana 709
UHiragana - Hierarchy List. 709
UHiragana - Member Functions and Data by Group 709
UHiragana - Enumerations 709
UHiragana - Inherited Member Functions and Data 709
UIdeographicAnnotation 709
UIdeographicAnnotation - Hierarchy List . . . 710
UIdeographicAnnotation - Member Functions and Data by Group 710
UIdeographicAnnotation - Enumerations . . . 710
UIdeographicAnnotation - Inherited Member Functions and Data 710
UIdeographicTelegraph 710
UIdeographicTelegraph - Hierarchy List . . . 710
UIdeographicTelegraph - Member Functions and Data by Group 710
UIdeographicTelegraph - Enumerations. . . . 710
UIdeographicTelegraph - Inherited Member Functions and Data 710
UKangXiRadical 711
UKangXiRadical - Hierarchy List 711
UKangXiRadical - Member Functions and Data by Group 711
UKangXiRadical - Enumerations 711
UKangXiRadical - Inherited Member Functions and Data 711
UKannada 711
UKannada - Hierarchy List 711
UKannada - Member Functions and Data by Group 711
UKannada - Enumerations 711
UKannada - Inherited Member Functions and Data 711
UKatakana 712
UKatakana - Hierarchy List. 712
UKatakana - Member Functions and Data by Group 712
UKatakana - Enumerations 712
UKatakana - Inherited Member Functions and Data 712
UKatakanaHiragana 712
UKatakanaHiragana - Hierarchy List 712
UKatakanaHiragana - Member Functions and Data by Group 712
UKatakanaHiragana - Enumerations. 712
UKatakanaHiragana - Inherited Member Functions and Data 712
ULao 713
ULao - Hierarchy List 713
ULao - Member Functions and Data by Group 713
ULao - Enumerations. 713
ULao - Inherited Member Functions and Data 713
ULatin 713
ULatin - Hierarchy List 713
ULatin - Member Functions and Data by Group 713
ULatin - Enumerations 713
ULatin - Inherited Member Functions and Data 713
ULatin1 714
ULatin1 - Hierarchy List. 714
ULatin1 - Member Functions and Data by Group 714

ULatin1 - Enumerations 714
ULatin1 - Inherited Member Functions and Data 714
ULatinSmallLigature 714
ULatinSmallLigature - Hierarchy List 714
ULatinSmallLigature - Member Functions and Data by Group 714
ULatinSmallLigature - Enumerations 714
ULatinSmallLigature - Inherited Member Functions and Data 714
ULetterLikeSymbol 715
ULetterLikeSymbol - Hierarchy List 715
ULetterLikeSymbol - Member Functions and Data by Group 715
ULetterLikeSymbol - Enumerations 715
ULetterLikeSymbol - Inherited Member Functions and Data 715
UMalayalam. 715
UMalayalam - Hierarchy List 715
UMalayalam - Member Functions and Data by Group 715
UMalayalam - Enumerations 715
UMalayalam - Inherited Member Functions and Data 715
UMathematicalOperators 716
UMathematicalOperators - Hierarchy List . . . 716
UMathematicalOperators - Member Functions and Data by Group 716
UMathematicalOperators - Enumerations . . . 716
UMathematicalOperators - Inherited Member Functions and Data 716
UMiscellaneousTechnical 716
UMiscellaneousTechnical - Hierarchy List . . . 716
UMiscellaneousTechnical - Member Functions and Data by Group 716
UMiscellaneousTechnical - Enumerations . . . 716
UMiscellaneousTechnical - Inherited Member Functions and Data 717
UModifierLetter 717
UModifierLetter - Hierarchy List 717
UModifierLetter - Member Functions and Data by Group. 717
UModifierLetter - Enumerations 717
UModifierLetter - Inherited Member Functions and Data 717
UOCR. 717
UOCR - Hierarchy List 717
UOCR - Member Functions and Data by Group 718
UOCR - Enumerations 718
UOCR - Inherited Member Functions and Data 718
UOriya 718
UOriya - Hierarchy List 718
UOriya - Member Functions and Data by Group 718
UOriya - Enumerations 718
UOriya - Inherited Member Functions and Data 718
UParenthesizedDigit 718
UParenthesizedDigit - Hierarchy List 719
UParenthesizedDigit - Member Functions and Data by Group 719
UParenthesizedDigit - Enumerations 719
UParenthesizedDigit - Inherited Member Functions and Data 719
UParenthesizedHangul 719
UParenthesizedHangul - Hierarchy List . . . 719
UParenthesizedHangul - Member Functions and Data by Group 719
UParenthesizedHangul - Enumerations. . . . 719
UParenthesizedHangul - Inherited Member Functions and Data 719
UParenthesizedIdeograph 720
UParenthesizedIdeograph - Hierarchy List. . . 720
UParenthesizedIdeograph - Member Functions and Data by Group 720
UParenthesizedIdeograph - Enumerations . . . 720
UParenthesizedIdeograph - Inherited Member Functions and Data 720
UParenthesizedLatin 720
UParenthesizedLatin - Hierarchy List 720
UParenthesizedLatin - Member Functions and Data by Group 720
UParenthesizedLatin - Enumerations 720
UParenthesizedLatin - Inherited Member Functions and Data 720
UParenthesizedNumber 721
UParenthesizedNumber - Hierarchy List . . . 721
UParenthesizedNumber - Member Functions and Data by Group 721
UParenthesizedNumber - Enumerations . . . 721
UParenthesizedNumber - Inherited Member Functions and Data 721
UPresentationFormForVertical 721
UPresentationFormForVertical - Hierarchy List 721
UPresentationFormForVertical - Member Functions and Data by Group 721
UPresentationFormForVertical - Enumerations 721
UPresentationFormForVertical - Inherited Member Functions and Data 722
URomanNumeral 722
URomanNumeral - Hierarchy List 722
URomanNumeral - Member Functions and Data by Group. 722
URomanNumeral - Enumerations 722
URomanNumeral - Inherited Member Functions and Data 722
USmallVariants. 722
USmallVariants - Hierarchy List 722
USmallVariants - Member Functions and Data by Group. 723
USmallVariants - Enumerations 723
USmallVariants - Inherited Member Functions and Data 723
UStandardPhonetic 723
UStandardPhonetic - Hierarchy List 723
UStandardPhonetic - Member Functions and Data by Group 723
UStandardPhonetic - Enumerations 723
UStandardPhonetic - Inherited Member Functions and Data 723
USubscript 723
USubscript - Hierarchy List. 723
USubscript - Member Functions and Data by Group 724
USubscript - Enumerations 724

USubscript - Inherited Member Functions and Data . . . 724
USuperscript . . . 724
USuperscript - Hierarchy List . . . 724
USuperscript - Member Functions and Data by Group . . . 724
USuperscript - Enumerations . . . 724
USuperscript - Inherited Member Functions and Data . . . 724
USyriac . . . 724
USyriac - Hierarchy List . . . 724
USyriac - Member Functions and Data by Group . . . 725
USyriac - Enumerations . . . 725
USyriac - Inherited Member Functions and Data 725
UTamil . . . 725
UTamil - Hierarchy List . . . 725
UTamil - Member Functions and Data by Group 725
UTamil - Enumerations . . . 725
UTamil - Inherited Member Functions and Data 725
UTelugu . . . 725
UTelugu - Hierarchy List . . . 725
UTelugu - Member Functions and Data by Group . . . 726
UTelugu - Enumerations. . . . 726
UTelugu - Inherited Member Functions and Data . . . 726
UThai . . . 726
UThai - Hierarchy List . . . 726
UThai - Member Functions and Data by Group 726
UThai - Enumerations . . . 726
UThai - Inherited Member Functions and Data 726
UUnicodeDigit . . . 726
UUnicodeDigit - Hierarchy List . . . 727
UUnicodeDigit - Member Functions and Data by Group. . . . 727
UUnicodeDigit - Enumerations . . . 727
UUnicodeDigit - Inherited Member Functions and Data . . . 727
UUnicodeSpecial . . . 727
UUnicodeSpecial - Hierarchy List. . . . 727
UUnicodeSpecial - Member Functions and Data by Group. . . . 727
UUnicodeSpecial - Enumerations . . . 727
UUnicodeSpecial - Inherited Member Functions and Data . . . 727
UVulgarFraction . . . 728
UVulgarFraction - Hierarchy List . . . 728
UVulgarFraction - Member Functions and Data by Group. . . . 728
UVulgarFraction - Enumerations . . . 728
UVulgarFraction - Inherited Member Functions and Data . . . 728
UWhiteSpace . . . 728
UWhiteSpace - Hierarchy List . . . 728
UWhiteSpace - Member Functions and Data by Group . . . 728
UWhiteSpace - Enumerations . . . 728
UWhiteSpace - Inherited Member Functions and Data . . . 728
UZapfDingbats . . . 729
UZapfDingbats - Hierarchy List . . . 729
UZapfDingbats - Member Functions and Data by Group. . . . 729
UZapfDingbats - Enumerations . . . 729
UZapfDingbats - Inherited Member Functions and Data . . . 729

Chapter 5. Error Handling, Tracing, and Testing Classes . . . 731

IAccessError . . . 731
IAccessError - Hierarchy List . . . 731
IAccessError - Member Functions and Data by Group . . . 731
IAccessError - Inherited Member Functions and Data . . . 733
IAlreadyExists . . . 733
IAlreadyExists - Hierarchy List . . . 733
IAlreadyExists - Member Functions and Data by Group . . . 734
IAlreadyExists - Inherited Member Functions and Data . . . 735
IArgumentDictionary . . . 735
IArgumentDictionary - Hierarchy List . . . 736
IArgumentDictionary - Member Functions and Data by Group . . . 736
IArgumentDictionary - Inherited Member Functions and Data . . . 742
IAssertionFailure . . . 742
IAssertionFailure - Hierarchy List . . . 742
IAssertionFailure - Member Functions and Data by Group. . . . 742
IAssertionFailure - Inherited Member Functions and Data . . . 744
IBaseErrorInfo . . . 745
IBaseErrorInfo - Hierarchy List . . . 746
IBaseErrorInfo - Member Functions and Data by Group . . . 746
IBaseErrorInfo - Enumerations. . . . 748
IBaseErrorInfo - Inherited Member Functions and Data . . . 748
ICannotProceed . . . 749
ICannotProceed - Hierarchy List . . . 749
ICannotProceed - Member Functions and Data by Group. . . . 749
ICannotProceed - Inherited Member Functions and Data . . . 750
ICLibErrorInfo . . . 751
ICLibErrorInfo - Hierarchy List . . . 751
ICLibErrorInfo - Member Functions and Data by Group . . . 752
ICLibErrorInfo - Inherited Member Functions and Data . . . 754
IComparisonTestOf . . . 754
IComparisonTestOf - Hierarchy List . . . 755
IComparisonTestOf - Member Functions and Data by Group . . . 755
IComparisonTestOf - Inherited Member Functions and Data . . . 757
IConditionInvalid . . . 758
IConditionInvalid - Hierarchy List . . . 758

IConditionInvalid - Member Functions and Data by Group . . . 758
IConditionInvalid - Inherited Member Functions and Data . . . 759
IDeviceError . . . 760
IDeviceError - Hierarchy List . . . 760
IDeviceError - Member Functions and Data by Group . . . 760
IDeviceError - Inherited Member Functions and Data . . . 761
IException . . . 762
IException - Hierarchy List . . . 764
IException - Member Functions and Data by Group . . . 765
IException - Enumerations . . . 771
IException - Type Definitions . . . 772
IException - Inherited Member Functions and Data . . . 772
IException::TraceFn . . . 772
IException::TraceFn - Hierarchy List . . . 772
IException::TraceFn - Member Functions and Data by Group . . . 773
IException::TraceFn - Inherited Member Functions and Data . . . 773
IExceptionLocation . . . 774
IExceptionLocation - Hierarchy List . . . 774
IExceptionLocation - Member Functions and Data by Group . . . 774
IExceptionLocation - Inherited Member Functions and Data . . . 776
IInvalidName . . . 776
IInvalidName - Hierarchy List . . . 776
IInvalidName - Member Functions and Data by Group . . . 776
IInvalidName - Inherited Member Functions and Data . . . 777
IInvalidParameter . . . 778
IInvalidParameter - Hierarchy List . . . 778
IInvalidParameter - Member Functions and Data by Group . . . 778
IInvalidParameter - Inherited Member Functions and Data . . . 780
IInvalidRequest . . . 780
IInvalidRequest - Hierarchy List . . . 781
IInvalidRequest - Member Functions and Data by Group . . . 781
IInvalidRequest - Inherited Member Functions and Data . . . 782
IMessageText . . . 783
IMessageText - Hierarchy List . . . 784
IMessageText - Member Functions and Data by Group . . . 784
IMessageText - Inherited Member Functions and Data . . . 788
IMustBeEmpty . . . 788
IMustBeEmpty - Hierarchy List . . . 788
IMustBeEmpty - Member Functions and Data by Group . . . 788
IMustBeEmpty - Inherited Member Functions and Data . . . 789
IObjectNotFound . . . 790
IObjectNotFound - Hierarchy List . . . 790
IObjectNotFound - Member Functions and Data by Group . . . 790
IObjectNotFound - Inherited Member Functions and Data . . . 791
IOutOfMemory . . . 792
IOutOfMemory - Hierarchy List . . . 792
IOutOfMemory - Member Functions and Data by Group . . . 792
IOutOfMemory - Inherited Member Functions and Data . . . 794
IOutOfSystemResource . . . 795
IOutOfSystemResource - Hierarchy List . . . 795
IOutOfSystemResource - Member Functions and Data by Group . . . 795
IOutOfSystemResource - Inherited Member Functions and Data . . . 797
IResourceExhausted . . . 797
IResourceExhausted - Hierarchy List . . . 798
IResourceExhausted - Member Functions and Data by Group . . . 798
IResourceExhausted - Inherited Member Functions and Data . . . 799
IStartStopTimingTest . . . 800
IStartStopTimingTest - Hierarchy List . . . 800
IStartStopTimingTest - Member Functions and Data by Group . . . 801
IStartStopTimingTest - Inherited Member Functions and Data . . . 803
IStreamTestOf . . . 805
IStreamTestOf - Hierarchy List . . . 805
IStreamTestOf - Member Functions and Data by Group . . . 805
IStreamTestOf - Inherited Member Functions and Data . . . 807
ISystemErrorInfo . . . 808
ISystemErrorInfo - Hierarchy List . . . 808
ISystemErrorInfo - Member Functions and Data by Group . . . 809
ISystemErrorInfo - Inherited Member Functions and Data . . . 811
ITest . . . 811
ITest - Hierarchy List . . . 811
ITest - Member Functions and Data by Group 812
ITest - Enumerations . . . 818
ITest - Inherited Member Functions and Data 818
ITestCollection . . . 819
ITestCollection - Hierarchy List . . . 819
ITestCollection - Member Functions and Data by Group . . . 819
ITestCollection - Inherited Member Functions and Data . . . 822
ITestMultiplexer . . . 823
ITestMultiplexer - Hierarchy List . . . 823
ITestMultiplexer - Member Functions and Data by Group . . . 824
ITestMultiplexer - Inherited Member Functions and Data . . . 827
ITieredTextBuffer . . . 828
ITieredTextBuffer - Hierarchy List . . . 828

ITieredTextBuffer - Member Functions and Data by Group 829
ITieredTextBuffer - Enumerations 833
ITieredTextBuffer - Type Definitions 833
ITieredTextBuffer - Inherited Member Functions and Data 833
ITimingTest 834
ITimingTest - Hierarchy List 834
ITimingTest - Member Functions and Data by Group 835
ITimingTest - Enumerations 842
ITimingTest - Inherited Member Functions and Data 843
ITimingTestStopwatch 843
ITimingTestStopwatch - Hierarchy List 843
ITimingTestStopwatch - Member Functions and Data by Group 844
ITimingTestStopwatch - Inherited Member Functions and Data 844
ITrace 845
ITrace - Hierarchy List 846
ITrace - Member Functions and Data by Group 846
ITrace - Enumerations 852
ITrace - Inherited Member Functions and Data 852

Chapter 6. Collection Classes 855

IABag 855
IABag - Hierarchy List 855
IABag - Member Functions and Data by Group 855
IABag - Inherited Member Functions and Data 861
IACollection 862
IACollection - Hierarchy List 862
IACollection - Member Functions and Data by Group 863
IACollection - Inherited Member Functions and Data 875
IACollectionBase 875
IACollectionBase - Hierarchy List. 875
IACollectionBase - Member Functions and Data by Group. 875
IACollectionBase - Inherited Member Functions and Data 876
IACollectionStreamer 876
IACollectionStreamer - Hierarchy List 876
IACollectionStreamer - Member Functions and Data by Group 876
IACollectionStreamer - Inherited Member Functions and Data 877
IADeque 877
IADeque - Hierarchy List 877
IADeque - Member Functions and Data by Group 878
IADeque - Inherited Member Functions and Data 881
IAEqualityCollection 882
IAEqualityCollection - Hierarchy List 883
IAEqualityCollection - Member Functions and Data by Group 883
IAEqualityCollection - Inherited Member Functions and Data 887
IAEqualityKeyCollection 888
IAEqualityKeyCollection - Hierarchy List . . . 888
IAEqualityKeyCollection - Member Functions and Data by Group 888
IAEqualityKeyCollection - Inherited Member Functions and Data 890
IAEqualityKeySortedCollection 891
IAEqualityKeySortedCollection - Hierarchy List 891
IAEqualityKeySortedCollection - Member Functions and Data by Group 892
IAEqualityKeySortedCollection - Inherited Member Functions and Data 893
IAEqualitySequence 894
IAEqualitySequence - Hierarchy List 894
IAEqualitySequence - Member Functions and Data by Group 895
IAEqualitySequence - Inherited Member Functions and Data 898
IAEqualitySortedCollection 899
IAEqualitySortedCollection - Hierarchy List . . 900
IAEqualitySortedCollection - Member Functions and Data by Group 900
IAEqualitySortedCollection - Inherited Member Functions and Data 901
IAHeap 902
IAHeap - Hierarchy List. 903
IAHeap - Member Functions and Data by Group 903
IAHeap - Inherited Member Functions and Data 904
IAKeyBag 905
IAKeyBag - Hierarchy List 905
IAKeyBag - Member Functions and Data by Group 906
IAKeyBag - Inherited Member Functions and Data 907
IAKeyCollection 908
IAKeyCollection - Hierarchy List 909
IAKeyCollection - Member Functions and Data by Group. 909
IAKeyCollection - Inherited Member Functions and Data 915
IAKeySet 916
IAKeySet - Hierarchy List 916
IAKeySet - Member Functions and Data by Group 917
IAKeySet - Inherited Member Functions and Data 919
IAKeySortedBag 921
IAKeySortedBag - Hierarchy List 921
IAKeySortedBag - Member Functions and Data by Group. 921
IAKeySortedBag - Inherited Member Functions and Data 923
IAKeySortedCollection 925
IAKeySortedCollection - Hierarchy List. . . . 925
IAKeySortedCollection - Member Functions and Data by Group 925
IAKeySortedCollection - Inherited Member Functions and Data 927
IAKeySortedSet 928
IAKeySortedSet - Hierarchy List 928

IAKeySortedSet - Member Functions and Data by Group 929
IAKeySortedSet - Inherited Member Functions and Data 932
IAMap 933
IAMap - Hierarchy List 934
IAMap - Member Functions and Data by Group 934
IAMap - Inherited Member Functions and Data 941
IAMultiwayTree 943
IAMultiwayTree - Hierarchy List 943
IAMultiwayTree - Member Functions and Data by Group 943
IAMultiwayTree - Inherited Member Functions and Data 947
IAOrderedCollection 948
IAOrderedCollection - Hierarchy List 948
IAOrderedCollection - Member Functions and Data by Group 949
IAOrderedCollection - Inherited Member Functions and Data 954
IApplicator 955
IApplicator - Hierarchy List 955
IApplicator - Member Functions and Data by Group 955
IApplicator - Inherited Member Functions and Data 956
IAPriorityQueue 956
IAPriorityQueue - Hierarchy List 956
IAPriorityQueue - Member Functions and Data by Group 957
IAPriorityQueue - Inherited Member Functions and Data 963
IAQueue 964
IAQueue - Hierarchy List 964
IAQueue - Member Functions and Data by Group 965
IAQueue - Inherited Member Functions and Data 968
IARelation 969
IARelation - Hierarchy List 970
IARelation - Member Functions and Data by Group 970
IARelation - Inherited Member Functions and Data 976
IARestrictedAccessCollection 978
IARestrictedAccessCollection - Hierarchy List 978
IARestrictedAccessCollection - Member Functions and Data by Group 978
IARestrictedAccessCollection - Inherited Member Functions and Data 990
IASequence 991
IASequence - Hierarchy List 991
IASequence - Member Functions and Data by Group 992
IASequence - Inherited Member Functions and Data 994
IASequentialCollection 995
IASequentialCollection - Hierarchy List. . . . 995
IASequentialCollection - Member Functions and Data by Group 996
IASequentialCollection - Inherited Member Functions and Data 1001
IASet 1003
IASet - Hierarchy List 1003
IASet - Member Functions and Data by Group 1003
IASet - Inherited Member Functions and Data 1009
IASortedBag 1011
IASortedBag - Hierarchy List 1011
IASortedBag - Member Functions and Data by Group 1011
IASortedBag - Inherited Member Functions and Data 1017
IASortedCollection 1019
IASortedCollection - Hierarchy List 1019
IASortedCollection - Member Functions and Data by Group 1019
IASortedCollection - Inherited Member Functions and Data 1021
IASortedMap 1022
IASortedMap - Hierarchy List 1022
IASortedMap - Member Functions and Data by Group 1023
IASortedMap - Inherited Member Functions and Data 1030
IASortedRelation 1031
IASortedRelation - Hierarchy List 1032
IASortedRelation - Member Functions and Data by Group 1032
IASortedRelation - Inherited Member Functions and Data 1038
IASortedSet 1040
IASortedSet - Hierarchy List 1040
IASortedSet - Member Functions and Data by Group 1041
IASortedSet - Inherited Member Functions and Data 1047
IAStack 1048
IAStack - Hierarchy List 1049
IAStack - Member Functions and Data by Group 1049
IAStack - Inherited Member Functions and Data 1053
IATree 1054
IATree - Hierarchy List 1054
IATree - Member Functions and Data by Group 1055
IATree - Inherited Member Functions and Data 1068
IATreeBase 1069
IATreeBase - Hierarchy List 1069
IATreeBase - Member Functions and Data by Group 1069
IATreeBase - Inherited Member Functions and Data 1070
IAutoElemPointer 1071
IAutoElemPointer - Hierarchy List 1071
IAutoElemPointer - Member Functions and Data by Group 1071
IAutoElemPointer - Inherited Member Functions and Data 1072
IAutoPointer 1072
IAutoPointer - Hierarchy List. 1072

IAutoPointer - Member Functions and Data by Group 1072
IAutoPointer - Inherited Member Functions and Data 1073
IBag 1074
IBag - Hierarchy List 1074
IBag - Member Functions and Data by Group 1074
IBag - Type Definitions 1074
IBag - Associated Globals 1075
IBag - Inherited Member Functions and Data 1075
IBagAsAvlTree 1077
IBagAsAvlTree - Hierarchy List 1077
IBagAsAvlTree - Member Functions and Data by Group 1077
IBagAsAvlTree - Type Definitions 1078
IBagAsAvlTree - Associated Globals 1078
IBagAsAvlTree - Inherited Member Functions and Data 1078
IBagAsBstTree 1080
IBagAsBstTree - Hierarchy List 1081
IBagAsBstTree - Member Functions and Data by Group 1081
IBagAsBstTree - Type Definitions 1081
IBagAsBstTree - Associated Globals. 1081
IBagAsBstTree - Inherited Member Functions and Data 1082
IBagAsDilTable 1084
IBagAsDilTable - Hierarchy List 1084
IBagAsDilTable - Member Functions and Data by Group 1084
IBagAsDilTable - Type Definitions 1084
IBagAsDilTable - Associated Globals 1085
IBagAsDilTable - Inherited Member Functions and Data 1085
IBagAsHshTable 1087
IBagAsHshTable - Hierarchy List 1087
IBagAsHshTable - Member Functions and Data by Group 1088
IBagAsHshTable - Type Definitions. 1088
IBagAsHshTable - Associated Globals 1088
IBagAsHshTable - Inherited Member Functions and Data 1088
IBagAsList 1090
IBagAsList - Hierarchy List 1091
IBagAsList - Member Functions and Data by Group 1091
IBagAsList - Type Definitions 1091
IBagAsList - Associated Globals 1092
IBagAsList - Inherited Member Functions and Data 1092
IBagAsTable 1094
IBagAsTable - Hierarchy List 1094
IBagAsTable - Member Functions and Data by Group 1094
IBagAsTable - Type Definitions 1095
IBagAsTable - Associated Globals 1095
IBagAsTable - Inherited Member Functions and Data 1095
ICollectionGuard 1097
ICollectionGuard - Hierarchy List 1098
ICollectionGuard - Member Functions and Data by Group 1098
ICollectionGuard - Inherited Member Functions and Data 1098
IConstantApplicator 1098
IConstantApplicator - Hierarchy List 1099
IConstantApplicator - Member Functions and Data by Group 1099
IConstantApplicator - Inherited Member Functions and Data 1099
ICursor 1099
ICursor - Hierarchy List 1100
ICursor - Member Functions and Data by Group 1100
ICursor - Inherited Member Functions and Data 1102
IDeque 1102
IDeque - Hierarchy List 1102
IDeque - Member Functions and Data by Group 1103
IDeque - Type Definitions 1103
IDeque - Associated Globals 1103
IDeque - Inherited Member Functions and Data 1103
IDequeAsDilTable 1105
IDequeAsDilTable - Hierarchy List 1105
IDequeAsDilTable - Member Functions and Data by Group 1106
IDequeAsDilTable - Type Definitions 1106
IDequeAsDilTable - Associated Globals . . . 1106
IDequeAsDilTable - Inherited Member Functions and Data 1107
IDequeAsList 1108
IDequeAsList - Hierarchy List 1109
IDequeAsList - Member Functions and Data by Group 1109
IDequeAsList - Type Definitions. 1109
IDequeAsList - Associated Globals 1109
IDequeAsList - Inherited Member Functions and Data 1110
IDequeAsTable 1111
IDequeAsTable - Hierarchy List 1112
IDequeAsTable - Member Functions and Data by Group 1112
IDequeAsTable - Type Definitions 1112
IDequeAsTable - Associated Globals 1112
IDequeAsTable - Inherited Member Functions and Data 1113
IElementCursor 1114
IElementCursor - Hierarchy List. 1115
IElementCursor - Member Functions and Data by Group 1116
IElementCursor - Inherited Member Functions and Data 1116
IElementTreeCursor 1117
IElementTreeCursor - Hierarchy List 1117
IElementTreeCursor - Member Functions and Data by Group 1117
IElementTreeCursor - Inherited Member Functions and Data 1118
IElemPointer 1118
IElemPointer - Hierarchy List. 1118

IElemPointer - Member Functions and Data by Group 1118
IElemPointer - Inherited Member Functions and Data 1119
IEqualitySequence 1119
IEqualitySequence - Hierarchy List 1120
IEqualitySequence - Member Functions and Data by Group 1120
IEqualitySequence - Type Definitions 1120
IEqualitySequence - Associated Globals . . . 1121
IEqualitySequence - Inherited Member Functions and Data 1121
IEqualitySequenceAsDilTable 1123
IEqualitySequenceAsDilTable - Hierarchy List 1123
IEqualitySequenceAsDilTable - Member Functions and Data by Group 1123
IEqualitySequenceAsDilTable - Type Definitions 1124
IEqualitySequenceAsDilTable - Associated Globals 1124
IEqualitySequenceAsDilTable - Inherited Member Functions and Data 1124
IEqualitySequenceAsList 1126
IEqualitySequenceAsList - Hierarchy List . . . 1127
IEqualitySequenceAsList - Member Functions and Data by Group 1127
IEqualitySequenceAsList - Type Definitions 1127
IEqualitySequenceAsList - Associated Globals 1127
IEqualitySequenceAsList - Inherited Member Functions and Data 1128
IEqualitySequenceAsTable 1130
IEqualitySequenceAsTable - Hierarchy List 1130
IEqualitySequenceAsTable - Member Functions and Data by Group 1130
IEqualitySequenceAsTable - Type Definitions 1130
IEqualitySequenceAsTable - Associated Globals 1131
IEqualitySequenceAsTable - Inherited Member Functions and Data 1131
IGBag 1133
IGBag - Hierarchy List 1133
IGBag - Member Functions and Data by Group 1133
IGBag - Type Definitions 1134
IGBag - Associated Globals 1135
IGBag - Inherited Member Functions and Data 1135
IGBagAsAvlTree 1137
IGBagAsAvlTree - Hierarchy List 1137
IGBagAsAvlTree - Member Functions and Data by Group 1137
IGBagAsAvlTree - Type Definitions 1138
IGBagAsAvlTree - Associated Globals 1139
IGBagAsAvlTree - Inherited Member Functions and Data 1139
IGBagAsAvlTreeCursor 1141
IGBagAsAvlTreeCursor - Hierarchy List . . . 1141
IGBagAsAvlTreeCursor - Member Functions and Data by Group 1141
IGBagAsAvlTreeCursor - Inherited Member Functions and Data 1142
IGBagAsBstTree 1143
IGBagAsBstTree - Hierarchy List. 1143
IGBagAsBstTree - Member Functions and Data by Group 1143
IGBagAsBstTree - Type Definitions 1144
IGBagAsBstTree - Associated Globals 1144
IGBagAsBstTree - Inherited Member Functions and Data 1145
IGBagAsBstTreeCursor 1146
IGBagAsBstTreeCursor - Hierarchy List . . . 1147
IGBagAsBstTreeCursor - Member Functions and Data by Group 1147
IGBagAsBstTreeCursor - Inherited Member Functions and Data 1147
IGBagAsDilTable 1148
IGBagAsDilTable - Hierarchy List 1148
IGBagAsDilTable - Member Functions and Data by Group 1148
IGBagAsDilTable - Type Definitions 1149
IGBagAsDilTable - Associated Globals 1150
IGBagAsDilTable - Inherited Member Functions and Data 1150
IGBagAsDilTableCursor 1152
IGBagAsDilTableCursor - Hierarchy List . . . 1152
IGBagAsDilTableCursor - Member Functions and Data by Group 1152
IGBagAsDilTableCursor - Inherited Member Functions and Data 1153
IGBagAsHshTable 1154
IGBagAsHshTable - Hierarchy List 1154
IGBagAsHshTable - Member Functions and Data by Group 1154
IGBagAsHshTable - Type Definitions 1155
IGBagAsHshTable - Associated Globals 1155
IGBagAsHshTable - Inherited Member Functions and Data 1156
IGBagAsHshTableCursor 1157
IGBagAsHshTableCursor - Hierarchy List . . . 1158
IGBagAsHshTableCursor - Member Functions and Data by Group 1158
IGBagAsHshTableCursor - Inherited Member Functions and Data 1158
IGBagAsList 1159
IGBagAsList - Hierarchy List 1159
IGBagAsList - Member Functions and Data by Group 1159
IGBagAsList - Type Definitions 1160
IGBagAsList - Associated Globals 1161
IGBagAsList - Inherited Member Functions and Data 1161
IGBagAsListCursor 1163
IGBagAsListCursor - Hierarchy List 1163
IGBagAsListCursor - Member Functions and Data by Group 1163
IGBagAsListCursor - Inherited Member Functions and Data 1164
IGBagAsTable 1164
IGBagAsTable - Hierarchy List 1165
IGBagAsTable - Member Functions and Data by Group 1165
IGBagAsTable - Type Definitions 1166
IGBagAsTable - Associated Globals 1166
IGBagAsTable - Inherited Member Functions and Data 1167
IGBagAsTableCursor 1168

IGBagAsTableCursor - Hierarchy List 1169
IGBagAsTableCursor - Member Functions and Data by Group 1169
IGBagAsTableCursor - Inherited Member Functions and Data 1169
IGBagCursor 1170
IGBagCursor - Hierarchy List. 1170
IGBagCursor - Member Functions and Data by Group 1170
IGBagCursor - Inherited Member Functions and Data 1171
IGDeque 1172
IGDeque - Hierarchy List 1172
IGDeque - Member Functions and Data by Group 1172
IGDeque - Type Definitions 1173
IGDeque - Associated Globals 1173
IGDeque - Inherited Member Functions and Data 1174
IGDequeAsDilTable 1175
IGDequeAsDilTable - Hierarchy List 1175
IGDequeAsDilTable - Member Functions and Data by Group 1175
IGDequeAsDilTable - Type Definitions. . . . 1176
IGDequeAsDilTable - Associated Globals . . . 1177
IGDequeAsDilTable - Inherited Member Functions and Data 1177
IGDequeAsDilTableCursor. 1179
IGDequeAsDilTableCursor - Hierarchy List 1179
IGDequeAsDilTableCursor - Member Functions and Data by Group 1179
IGDequeAsDilTableCursor - Inherited Member Functions and Data 1180
IGDequeAsList 1181
IGDequeAsList - Hierarchy List 1181
IGDequeAsList - Member Functions and Data by Group 1181
IGDequeAsList - Type Definitions 1182
IGDequeAsList - Associated Globals 1182
IGDequeAsList - Inherited Member Functions and Data 1183
IGDequeAsListCursor 1184
IGDequeAsListCursor - Hierarchy List . . . 1184
IGDequeAsListCursor - Member Functions and Data by Group 1184
IGDequeAsListCursor - Inherited Member Functions and Data 1185
IGDequeAsTable 1186
IGDequeAsTable - Hierarchy List 1186
IGDequeAsTable - Member Functions and Data by Group 1186
IGDequeAsTable - Type Definitions. 1187
IGDequeAsTable - Associated Globals 1188
IGDequeAsTable - Inherited Member Functions and Data 1188
IGDequeAsTableCursor. 1190
IGDequeAsTableCursor - Hierarchy List . . . 1190
IGDequeAsTableCursor - Member Functions and Data by Group 1190
IGDequeAsTableCursor - Inherited Member Functions and Data 1190
IGDequeCursor 1191
IGDequeCursor - Hierarchy List. 1191
IGDequeCursor - Member Functions and Data by Group 1192
IGDequeCursor - Inherited Member Functions and Data 1192
IGEqualitySequence 1193
IGEqualitySequence - Hierarchy List 1193
IGEqualitySequence - Member Functions and Data by Group 1193
IGEqualitySequence - Type Definitions . . . 1194
IGEqualitySequence - Associated Globals . . . 1195
IGEqualitySequence - Inherited Member Functions and Data 1195
IGEqualitySequenceAsDilTable 1197
IGEqualitySequenceAsDilTable - Hierarchy List 1197
IGEqualitySequenceAsDilTable - Member Functions and Data by Group 1197
IGEqualitySequenceAsDilTable - Type Definitions 1198
IGEqualitySequenceAsDilTable - Associated Globals 1199
IGEqualitySequenceAsDilTable - Inherited Member Functions and Data 1199
IGEqualitySequenceAsDilTableCursor 1201
IGEqualitySequenceAsDilTableCursor - Hierarchy List. 1201
IGEqualitySequenceAsDilTableCursor - Member Functions and Data by Group . . . 1201
IGEqualitySequenceAsDilTableCursor - Inherited Member Functions and Data . . . 1202
IGEqualitySequenceAsList. 1203
IGEqualitySequenceAsList - Hierarchy List 1203
IGEqualitySequenceAsList - Member Functions and Data by Group 1203
IGEqualitySequenceAsList - Type Definitions 1204
IGEqualitySequenceAsList - Associated Globals 1205
IGEqualitySequenceAsList - Inherited Member Functions and Data 1205
IGEqualitySequenceAsListCursor 1207
IGEqualitySequenceAsListCursor - Hierarchy List 1207
IGEqualitySequenceAsListCursor - Member Functions and Data by Group 1207
IGEqualitySequenceAsListCursor - Inherited Member Functions and Data 1208
IGEqualitySequenceAsTable 1209
IGEqualitySequenceAsTable - Hierarchy List 1209
IGEqualitySequenceAsTable - Member Functions and Data by Group 1209
IGEqualitySequenceAsTable - Type Definitions 1210
IGEqualitySequenceAsTable - Associated Globals 1210
IGEqualitySequenceAsTable - Inherited Member Functions and Data 1211
IGEqualitySequenceAsTableCursor 1212
IGEqualitySequenceAsTableCursor - Hierarchy List 1213
IGEqualitySequenceAsTableCursor - Member Functions and Data by Group 1213

IGEqualitySequenceAsTableCursor - Inherited Member Functions and Data 1213
IGEqualitySequenceCursor 1214
IGEqualitySequenceCursor - Hierarchy List 1214
IGEqualitySequenceCursor - Member Functions and Data by Group 1214
IGEqualitySequenceCursor - Inherited Member Functions and Data 1215
IGHeap 1216
IGHeap - Hierarchy List 1216
IGHeap - Member Functions and Data by Group 1216
IGHeap - Type Definitions 1217
IGHeap - Associated Globals 1218
IGHeap - Inherited Member Functions and Data 1218
IGHeapAsDilTable 1218
IGHeapAsDilTable - Hierarchy List. 1218
IGHeapAsDilTable - Member Functions and Data by Group 1219
IGHeapAsDilTable - Type Definitions 1219
IGHeapAsDilTable - Associated Globals . . . 1220
IGHeapAsDilTable - Inherited Member Functions and Data 1220
IGHeapAsDilTableCursor 1221
IGHeapAsDilTableCursor - Hierarchy List . . 1221
IGHeapAsDilTableCursor - Member Functions and Data by Group 1221
IGHeapAsDilTableCursor - Inherited Member Functions and Data 1221
IGHeapAsList 1222
IGHeapAsList - Hierarchy List 1222
IGHeapAsList - Member Functions and Data by Group 1222
IGHeapAsList - Type Definitions 1223
IGHeapAsList - Associated Globals. 1224
IGHeapAsList - Inherited Member Functions and Data 1224
IGHeapAsListCursor 1224
IGHeapAsListCursor - Hierarchy List 1225
IGHeapAsListCursor - Member Functions and Data by Group 1225
IGHeapAsListCursor - Inherited Member Functions and Data 1225
IGHeapAsTable 1226
IGHeapAsTable - Hierarchy List. 1226
IGHeapAsTable - Member Functions and Data by Group 1226
IGHeapAsTable - Type Definitions 1227
IGHeapAsTable - Associated Globals 1227
IGHeapAsTable - Inherited Member Functions and Data 1228
IGHeapAsTableCursor 1228
IGHeapAsTableCursor - Hierarchy List . . . 1228
IGHeapAsTableCursor - Member Functions and Data by Group 1228
IGHeapAsTableCursor - Inherited Member Functions and Data 1229
IGHeapCursor. 1230
IGHeapCursor - Hierarchy List 1230
IGHeapCursor - Member Functions and Data by Group 1230
IGHeapCursor - Inherited Member Functions and Data 1230
IGKeyBag 1231
IGKeyBag - Hierarchy List 1231
IGKeyBag - Member Functions and Data by Group 1231
IGKeyBag - Type Definitions 1232
IGKeyBag - Associated Globals 1233
IGKeyBag - Inherited Member Functions and Data 1233
IGKeyBagAsDilTable 1235
IGKeyBagAsDilTable - Hierarchy List 1235
IGKeyBagAsDilTable - Member Functions and Data by Group 1235
IGKeyBagAsDilTable - Type Definitions . . . 1236
IGKeyBagAsDilTable - Associated Globals . . 1236
IGKeyBagAsDilTable - Inherited Member Functions and Data 1237
IGKeyBagAsDilTableCursor 1239
IGKeyBagAsDilTableCursor - Hierarchy List 1239
IGKeyBagAsDilTableCursor - Member Functions and Data by Group 1239
IGKeyBagAsDilTableCursor - Inherited Member Functions and Data 1239
IGKeyBagAsHshTable 1240
IGKeyBagAsHshTable - Hierarchy List . . . 1240
IGKeyBagAsHshTable - Member Functions and Data by Group 1240
IGKeyBagAsHshTable - Type Definitions . . . 1241
IGKeyBagAsHshTable - Associated Globals 1242
IGKeyBagAsHshTable - Inherited Member Functions and Data 1242
IGKeyBagAsHshTableCursor 1244
IGKeyBagAsHshTableCursor - Hierarchy List 1244
IGKeyBagAsHshTableCursor - Member Functions and Data by Group 1244
IGKeyBagAsHshTableCursor - Inherited Member Functions and Data 1245
IGKeyBagAsList 1245
IGKeyBagAsList - Hierarchy List 1246
IGKeyBagAsList - Member Functions and Data by Group 1246
IGKeyBagAsList - Type Definitions. 1247
IGKeyBagAsList - Associated Globals 1247
IGKeyBagAsList - Inherited Member Functions and Data 1248
IGKeyBagAsListCursor. 1249
IGKeyBagAsListCursor - Hierarchy List . . . 1250
IGKeyBagAsListCursor - Member Functions and Data by Group 1250
IGKeyBagAsListCursor - Inherited Member Functions and Data 1250
IGKeyBagAsTable 1251
IGKeyBagAsTable - Hierarchy List 1251
IGKeyBagAsTable - Member Functions and Data by Group 1251
IGKeyBagAsTable - Type Definitions 1252
IGKeyBagAsTable - Associated Globals . . . 1253

IGKeyBagAsTable - Inherited Member Functions and Data 1253
IGKeyBagAsTableCursor 1255
IGKeyBagAsTableCursor - Hierarchy List. . . 1255
IGKeyBagAsTableCursor - Member Functions and Data by Group 1255
IGKeyBagAsTableCursor - Inherited Member Functions and Data 1256
IGKeyBagCursor 1256
IGKeyBagCursor - Hierarchy List 1256
IGKeyBagCursor - Member Functions and Data by Group 1257
IGKeyBagCursor - Inherited Member Functions and Data 1257
IGKeySet 1258
IGKeySet - Hierarchy List 1258
IGKeySet - Member Functions and Data by Group 1258
IGKeySet - Type Definitions 1259
IGKeySet - Associated Globals 1260
IGKeySet - Inherited Member Functions and Data 1260
IGKeySetAsAvlTree 1262
IGKeySetAsAvlTree - Hierarchy List 1262
IGKeySetAsAvlTree - Member Functions and Data by Group 1262
IGKeySetAsAvlTree - Type Definitions. . . . 1263
IGKeySetAsAvlTree - Associated Globals . . . 1264
IGKeySetAsAvlTree - Inherited Member Functions and Data 1264
IGKeySetAsAvlTreeCursor. 1266
IGKeySetAsAvlTreeCursor - Hierarchy List 1266
IGKeySetAsAvlTreeCursor - Member Functions and Data by Group 1266
IGKeySetAsAvlTreeCursor - Inherited Member Functions and Data 1267
IGKeySetAsBstTree 1267
IGKeySetAsBstTree - Hierarchy List 1267
IGKeySetAsBstTree - Member Functions and Data by Group 1268
IGKeySetAsBstTree - Type Definitions 1268
IGKeySetAsBstTree - Associated Globals . . . 1269
IGKeySetAsBstTree - Inherited Member Functions and Data 1269
IGKeySetAsBstTreeCursor 1271
IGKeySetAsBstTreeCursor - Hierarchy List 1271
IGKeySetAsBstTreeCursor - Member Functions and Data by Group 1272
IGKeySetAsBstTreeCursor - Inherited Member Functions and Data 1272
IGKeySetAsDilTable. 1273
IGKeySetAsDilTable - Hierarchy List 1273
IGKeySetAsDilTable - Member Functions and Data by Group 1273
IGKeySetAsDilTable - Type Definitions . . . 1274
IGKeySetAsDilTable - Associated Globals. . . 1275
IGKeySetAsDilTable - Inherited Member Functions and Data 1275
IGKeySetAsDilTableCursor 1277
IGKeySetAsDilTableCursor - Hierarchy List 1277
IGKeySetAsDilTableCursor - Member Functions and Data by Group 1277
IGKeySetAsDilTableCursor - Inherited Member Functions and Data 1278
IGKeySetAsHshTable 1278
IGKeySetAsHshTable - Hierarchy List 1278
IGKeySetAsHshTable - Member Functions and Data by Group 1279
IGKeySetAsHshTable - Type Definitions . . . 1279
IGKeySetAsHshTable - Associated Globals 1280
IGKeySetAsHshTable - Inherited Member Functions and Data 1280
IGKeySetAsHshTableCursor 1282
IGKeySetAsHshTableCursor - Hierarchy List 1282
IGKeySetAsHshTableCursor - Member Functions and Data by Group 1283
IGKeySetAsHshTableCursor - Inherited Member Functions and Data 1283
IGKeySetAsList 1284
IGKeySetAsList - Hierarchy List. 1284
IGKeySetAsList - Member Functions and Data by Group 1284
IGKeySetAsList - Type Definitions 1285
IGKeySetAsList - Associated Globals 1286
IGKeySetAsList - Inherited Member Functions and Data 1286
IGKeySetAsListCursor 1288
IGKeySetAsListCursor - Hierarchy List . . . 1288
IGKeySetAsListCursor - Member Functions and Data by Group 1288
IGKeySetAsListCursor - Inherited Member Functions and Data 1289
IGKeySetAsTable. 1289
IGKeySetAsTable - Hierarchy List 1289
IGKeySetAsTable - Member Functions and Data by Group 1290
IGKeySetAsTable - Type Definitions 1290
IGKeySetAsTable - Associated Globals. . . . 1291
IGKeySetAsTable - Inherited Member Functions and Data 1291
IGKeySetAsTableCursor 1293
IGKeySetAsTableCursor - Hierarchy List . . . 1293
IGKeySetAsTableCursor - Member Functions and Data by Group 1294
IGKeySetAsTableCursor - Inherited Member Functions and Data 1294
IGKeySetCursor 1295
IGKeySetCursor - Hierarchy List 1295
IGKeySetCursor - Member Functions and Data by Group 1295
IGKeySetCursor - Inherited Member Functions and Data 1296
IGKeySortedBag 1296
IGKeySortedBag - Hierarchy List 1297
IGKeySortedBag - Member Functions and Data by Group 1297
IGKeySortedBag - Type Definitions. 1297
IGKeySortedBag - Associated Globals 1298
IGKeySortedBag - Inherited Member Functions and Data 1299
IGKeySortedBagAsDilTable 1301

IGKeySortedBagAsDilTable - Hierarchy List 1301
IGKeySortedBagAsDilTable - Member Functions and Data by Group 1301
IGKeySortedBagAsDilTable - Type Definitions 1302
IGKeySortedBagAsDilTable - Associated Globals 1302
IGKeySortedBagAsDilTable - Inherited Member Functions and Data 1303
IGKeySortedBagAsDilTableCursor 1305
IGKeySortedBagAsDilTableCursor - Hierarchy List 1305
IGKeySortedBagAsDilTableCursor - Member Functions and Data by Group 1305
IGKeySortedBagAsDilTableCursor - Inherited Member Functions and Data 1306
IGKeySortedBagAsList 1307
IGKeySortedBagAsList - Hierarchy List . . . 1307
IGKeySortedBagAsList - Member Functions and Data by Group 1307
IGKeySortedBagAsList - Type Definitions . . 1308
IGKeySortedBagAsList - Associated Globals 1309
IGKeySortedBagAsList - Inherited Member Functions and Data 1309
IGKeySortedBagAsListCursor 1311
IGKeySortedBagAsListCursor - Hierarchy List 1311
IGKeySortedBagAsListCursor - Member Functions and Data by Group 1311
IGKeySortedBagAsListCursor - Inherited Member Functions and Data 1312
IGKeySortedBagAsTable 1313
IGKeySortedBagAsTable - Hierarchy List . . . 1313
IGKeySortedBagAsTable - Member Functions and Data by Group 1313
IGKeySortedBagAsTable - Type Definitions 1314
IGKeySortedBagAsTable - Associated Globals 1315
IGKeySortedBagAsTable - Inherited Member Functions and Data 1315
IGKeySortedBagAsTableCursor 1317
IGKeySortedBagAsTableCursor - Hierarchy List 1317
IGKeySortedBagAsTableCursor - Member Functions and Data by Group 1318
IGKeySortedBagAsTableCursor - Inherited Member Functions and Data 1318
IGKeySortedBagCursor. 1319
IGKeySortedBagCursor - Hierarchy List . . . 1319
IGKeySortedBagCursor - Member Functions and Data by Group 1319
IGKeySortedBagCursor - Inherited Member Functions and Data 1320
IGKeySortedSet 1321
IGKeySortedSet - Hierarchy List. 1321
IGKeySortedSet - Member Functions and Data by Group 1321
IGKeySortedSet - Type Definitions 1322
IGKeySortedSet - Associated Globals 1323
IGKeySortedSet - Inherited Member Functions and Data 1323
IGKeySortedSetAsAvlTree 1325
IGKeySortedSetAsAvlTree - Hierarchy List 1325
IGKeySortedSetAsAvlTree - Member Functions and Data by Group 1326
IGKeySortedSetAsAvlTree - Type Definitions 1326
IGKeySortedSetAsAvlTree - Associated Globals 1327
IGKeySortedSetAsAvlTree - Inherited Member Functions and Data 1327
IGKeySortedSetAsAvlTreeCursor 1329
IGKeySortedSetAsAvlTreeCursor - Hierarchy List 1330
IGKeySortedSetAsAvlTreeCursor - Member Functions and Data by Group 1330
IGKeySortedSetAsAvlTreeCursor - Inherited Member Functions and Data 1330
IGKeySortedSetAsBstTree 1331
IGKeySortedSetAsBstTree - Hierarchy List . . 1331
IGKeySortedSetAsBstTree - Member Functions and Data by Group 1332
IGKeySortedSetAsBstTree - Type Definitions 1332
IGKeySortedSetAsBstTree - Associated Globals 1333
IGKeySortedSetAsBstTree - Inherited Member Functions and Data 1333
IGKeySortedSetAsBstTreeCursor. 1335
IGKeySortedSetAsBstTreeCursor - Hierarchy List 1336
IGKeySortedSetAsBstTreeCursor - Member Functions and Data by Group 1336
IGKeySortedSetAsBstTreeCursor - Inherited Member Functions and Data 1336
IGKeySortedSetAsDilTable 1337
IGKeySortedSetAsDilTable - Hierarchy List 1337
IGKeySortedSetAsDilTable - Member Functions and Data by Group 1338
IGKeySortedSetAsDilTable - Type Definitions 1338
IGKeySortedSetAsDilTable - Associated Globals 1339
IGKeySortedSetAsDilTable - Inherited Member Functions and Data 1339
IGKeySortedSetAsDilTableCursor 1341
IGKeySortedSetAsDilTableCursor - Hierarchy List 1342
IGKeySortedSetAsDilTableCursor - Member Functions and Data by Group 1342
IGKeySortedSetAsDilTableCursor - Inherited Member Functions and Data 1342
IGKeySortedSetAsList 1343
IGKeySortedSetAsList - Hierarchy List . . . 1343
IGKeySortedSetAsList - Member Functions and Data by Group 1344
IGKeySortedSetAsList - Type Definitions . . . 1344
IGKeySortedSetAsList - Associated Globals 1345
IGKeySortedSetAsList - Inherited Member Functions and Data 1345
IGKeySortedSetAsListCursor 1347
IGKeySortedSetAsListCursor - Hierarchy List 1348
IGKeySortedSetAsListCursor - Member Functions and Data by Group 1348
IGKeySortedSetAsListCursor - Inherited Member Functions and Data 1348
IGKeySortedSetAsTable 1349
IGKeySortedSetAsTable - Hierarchy List . . . 1349
IGKeySortedSetAsTable - Member Functions and Data by Group 1350
IGKeySortedSetAsTable - Type Definitions 1350
IGKeySortedSetAsTable - Associated Globals 1351

IGKeySortedSetAsTable - Inherited Member Functions and Data 1351
IGKeySortedSetAsTableCursor 1353
IGKeySortedSetAsTableCursor - Hierarchy List 1354
IGKeySortedSetAsTableCursor - Member Functions and Data by Group 1354
IGKeySortedSetAsTableCursor - Inherited Member Functions and Data 1354
IGKeySortedSetCursor 1355
IGKeySortedSetCursor - Hierarchy List . . . 1355
IGKeySortedSetCursor - Member Functions and Data by Group 1355
IGKeySortedSetCursor - Inherited Member Functions and Data 1356
IGMap 1357
IGMap - Hierarchy List. 1357
IGMap - Member Functions and Data by Group 1357
IGMap - Type Definitions 1358
IGMap - Associated Globals 1359
IGMap - Inherited Member Functions and Data 1359
IGMapAsAvlTree. 1361
IGMapAsAvlTree - Hierarchy List 1361
IGMapAsAvlTree - Member Functions and Data by Group 1362
IGMapAsAvlTree - Type Definitions 1362
IGMapAsAvlTree - Associated Globals. . . . 1363
IGMapAsAvlTree - Inherited Member Functions and Data 1363
IGMapAsAvlTreeCursor 1365
IGMapAsAvlTreeCursor - Hierarchy List . . . 1366
IGMapAsAvlTreeCursor - Member Functions and Data by Group 1366
IGMapAsAvlTreeCursor - Inherited Member Functions and Data 1366
IGMapAsBstTree 1367
IGMapAsBstTree - Hierarchy List 1367
IGMapAsBstTree - Member Functions and Data by Group 1367
IGMapAsBstTree - Type Definitions 1368
IGMapAsBstTree - Associated Globals. . . . 1369
IGMapAsBstTree - Inherited Member Functions and Data 1369
IGMapAsBstTreeCursor 1371
IGMapAsBstTreeCursor - Hierarchy List . . . 1371
IGMapAsBstTreeCursor - Member Functions and Data by Group 1371
IGMapAsBstTreeCursor - Inherited Member Functions and Data 1372
IGMapAsDilTable 1373
IGMapAsDilTable - Hierarchy List 1373
IGMapAsDilTable - Member Functions and Data by Group 1373
IGMapAsDilTable - Type Definitions 1374
IGMapAsDilTable - Associated Globals . . . 1374
IGMapAsDilTable - Inherited Member Functions and Data 1375
IGMapAsDilTableCursor 1377
IGMapAsDilTableCursor - Hierarchy List. . . 1377
IGMapAsDilTableCursor - Member Functions and Data by Group 1377
IGMapAsDilTableCursor - Inherited Member Functions and Data 1378
IGMapAsHshTable 1378
IGMapAsHshTable - Hierarchy List 1379
IGMapAsHshTable - Member Functions and Data by Group 1379
IGMapAsHshTable - Type Definitions 1380
IGMapAsHshTable - Associated Globals . . . 1380
IGMapAsHshTable - Inherited Member Functions and Data 1381
IGMapAsHshTableCursor 1383
IGMapAsHshTableCursor - Hierarchy List 1383
IGMapAsHshTableCursor - Member Functions and Data by Group 1383
IGMapAsHshTableCursor - Inherited Member Functions and Data 1384
IGMapAsList 1384
IGMapAsList - Hierarchy List 1384
IGMapAsList - Member Functions and Data by Group 1385
IGMapAsList - Type Definitions. 1386
IGMapAsList - Associated Globals 1386
IGMapAsList - Inherited Member Functions and Data 1387
IGMapAsListCursor. 1389
IGMapAsListCursor - Hierarchy List 1389
IGMapAsListCursor - Member Functions and Data by Group 1389
IGMapAsListCursor - Inherited Member Functions and Data 1390
IGMapAsTable 1390
IGMapAsTable - Hierarchy List 1390
IGMapAsTable - Member Functions and Data by Group 1391
IGMapAsTable - Type Definitions 1391
IGMapAsTable - Associated Globals 1392
IGMapAsTable - Inherited Member Functions and Data 1392
IGMapAsTableCursor 1395
IGMapAsTableCursor - Hierarchy List. . . . 1395
IGMapAsTableCursor - Member Functions and Data by Group 1395
IGMapAsTableCursor - Inherited Member Functions and Data 1395
IGMapCursor 1396
IGMapCursor - Hierarchy List 1396
IGMapCursor - Member Functions and Data by Group 1396
IGMapCursor - Inherited Member Functions and Data 1397
IGMultiwayTree 1398
IGMultiwayTree - Hierarchy List 1398
IGMultiwayTree - Member Functions and Data by Group 1398
IGMultiwayTree - Type Definitions. 1398
IGMultiwayTree - Associated Globals 1399
IGMultiwayTree - Inherited Member Functions and Data 1399
IGMultiwayTreeCursor. 1401
IGMultiwayTreeCursor - Hierarchy List . . . 1401

IGMultiwayTreeCursor - Member Functions and Data by Group 1401
IGMultiwayTreeCursor - Inherited Member Functions and Data 1402
IGPriorityQueue 1402
IGPriorityQueue - Hierarchy List 1403
IGPriorityQueue - Member Functions and Data by Group 1403
IGPriorityQueue - Type Definitions. 1404
IGPriorityQueue - Associated Globals 1404
IGPriorityQueue - Inherited Member Functions and Data 1405
IGPriorityQueueAsDilTable 1406
IGPriorityQueueAsDilTable - Hierarchy List 1406
IGPriorityQueueAsDilTable - Member Functions and Data by Group 1407
IGPriorityQueueAsDilTable - Type Definitions 1407
IGPriorityQueueAsDilTable - Associated Globals 1408
IGPriorityQueueAsDilTable - Inherited Member Functions and Data 1408
IGPriorityQueueAsDilTableCursor 1410
IGPriorityQueueAsDilTableCursor - Hierarchy List 1410
IGPriorityQueueAsDilTableCursor - Member Functions and Data by Group 1410
IGPriorityQueueAsDilTableCursor - Inherited Member Functions and Data 1411
IGPriorityQueueAsList 1412
IGPriorityQueueAsList - Hierarchy List . . . 1412
IGPriorityQueueAsList - Member Functions and Data by Group 1412
IGPriorityQueueAsList - Type Definitions . . 1413
IGPriorityQueueAsList - Associated Globals 1414
IGPriorityQueueAsList - Inherited Member Functions and Data 1414
IGPriorityQueueAsListCursor 1416
IGPriorityQueueAsListCursor - Hierarchy List 1416
IGPriorityQueueAsListCursor - Member Functions and Data by Group 1416
IGPriorityQueueAsListCursor - Inherited Member Functions and Data 1417
IGPriorityQueueAsTable 1418
IGPriorityQueueAsTable - Hierarchy List. . . 1418
IGPriorityQueueAsTable - Member Functions and Data by Group 1418
IGPriorityQueueAsTable - Type Definitions 1419
IGPriorityQueueAsTable - Associated Globals 1419
IGPriorityQueueAsTable - Inherited Member Functions and Data 1420
IGPriorityQueueAsTableCursor 1421
IGPriorityQueueAsTableCursor - Hierarchy List 1421
IGPriorityQueueAsTableCursor - Member Functions and Data by Group 1422
IGPriorityQueueAsTableCursor - Inherited Member Functions and Data 1422
IGPriorityQueueCursor. 1423
IGPriorityQueueCursor - Hierarchy List . . . 1423
IGPriorityQueueCursor - Member Functions and Data by Group 1423
IGPriorityQueueCursor - Inherited Member Functions and Data 1424
IGQueue 1425
IGQueue - Hierarchy List 1425
IGQueue - Member Functions and Data by Group 1425
IGQueue - Type Definitions 1426
IGQueue - Associated Globals 1426
IGQueue - Inherited Member Functions and Data 1427
IGQueueAsDilTable 1428
IGQueueAsDilTable - Hierarchy List 1429
IGQueueAsDilTable - Member Functions and Data by Group 1429
IGQueueAsDilTable - Type Definitions . . . 1430
IGQueueAsDilTable - Associated Globals. . . 1430
IGQueueAsDilTable - Inherited Member Functions and Data 1431
IGQueueAsDilTableCursor 1432
IGQueueAsDilTableCursor - Hierarchy List 1432
IGQueueAsDilTableCursor - Member Functions and Data by Group 1432
IGQueueAsDilTableCursor - Inherited Member Functions and Data 1433
IGQueueAsList 1434
IGQueueAsList - Hierarchy List 1434
IGQueueAsList - Member Functions and Data by Group 1434
IGQueueAsList - Type Definitions 1435
IGQueueAsList - Associated Globals 1436
IGQueueAsList - Inherited Member Functions and Data 1436
IGQueueAsListCursor 1438
IGQueueAsListCursor - Hierarchy List . . . 1438
IGQueueAsListCursor - Member Functions and Data by Group 1438
IGQueueAsListCursor - Inherited Member Functions and Data 1438
IGQueueAsTable 1439
IGQueueAsTable - Hierarchy List 1439
IGQueueAsTable - Member Functions and Data by Group 1440
IGQueueAsTable - Type Definitions 1440
IGQueueAsTable - Associated Globals. . . . 1441
IGQueueAsTable - Inherited Member Functions and Data 1441
IGQueueAsTableCursor 1443
IGQueueAsTableCursor - Hierarchy List . . . 1443
IGQueueAsTableCursor - Member Functions and Data by Group 1443
IGQueueAsTableCursor - Inherited Member Functions and Data 1444
IGQueueCursor 1445
IGQueueCursor - Hierarchy List. 1445
IGQueueCursor - Member Functions and Data by Group 1445
IGQueueCursor - Inherited Member Functions and Data 1445
IGRelation 1447
IGRelation - Hierarchy List 1447

IGRelation - Member Functions and Data by Group 1447
IGRelation - Type Definitions. 1448
IGRelation - Associated Globals 1448
IGRelation - Inherited Member Functions and Data 1449
IGRelationAsDilTable 1451
IGRelationAsDilTable - Hierarchy List. . . . 1451
IGRelationAsDilTable - Member Functions and Data by Group 1451
IGRelationAsDilTable - Type Definitions . . . 1452
IGRelationAsDilTable - Associated Globals 1452
IGRelationAsDilTable - Inherited Member Functions and Data 1453
IGRelationAsDilTableCursor 1455
IGRelationAsDilTableCursor - Hierarchy List 1455
IGRelationAsDilTableCursor - Member Functions and Data by Group 1455
IGRelationAsDilTableCursor - Inherited Member Functions and Data 1456
IGRelationAsHshTable 1456
IGRelationAsHshTable - Hierarchy List . . . 1456
IGRelationAsHshTable - Member Functions and Data by Group 1457
IGRelationAsHshTable - Type Definitions. . . 1457
IGRelationAsHshTable - Associated Globals 1458
IGRelationAsHshTable - Inherited Member Functions and Data 1458
IGRelationAsHshTableCursor. 1460
IGRelationAsHshTableCursor - Hierarchy List 1461
IGRelationAsHshTableCursor - Member Functions and Data by Group 1461
IGRelationAsHshTableCursor - Inherited Member Functions and Data 1461
IGRelationAsList 1462
IGRelationAsList - Hierarchy List 1462
IGRelationAsList - Member Functions and Data by Group 1462
IGRelationAsList - Type Definitions 1463
IGRelationAsList - Associated Globals. . . . 1464
IGRelationAsList - Inherited Member Functions and Data 1464
IGRelationAsListCursor 1466
IGRelationAsListCursor - Hierarchy List . . . 1466
IGRelationAsListCursor - Member Functions and Data by Group 1466
IGRelationAsListCursor - Inherited Member Functions and Data 1467
IGRelationAsTable 1468
IGRelationAsTable - Hierarchy List. 1468
IGRelationAsTable - Member Functions and Data by Group 1468
IGRelationAsTable - Type Definitions 1469
IGRelationAsTable - Associated Globals . . . 1469
IGRelationAsTable - Inherited Member Functions and Data 1470
IGRelationAsTableCursor 1472
IGRelationAsTableCursor - Hierarchy List . . 1472
IGRelationAsTableCursor - Member Functions and Data by Group 1472
IGRelationAsTableCursor - Inherited Member Functions and Data 1473
IGRelationCursor. 1473
IGRelationCursor - Hierarchy List 1473
IGRelationCursor - Member Functions and Data by Group 1474
IGRelationCursor - Inherited Member Functions and Data 1474
IGSequence. 1475
IGSequence - Hierarchy List 1475
IGSequence - Member Functions and Data by Group 1475
IGSequence - Type Definitions 1476
IGSequence - Associated Globals 1476
IGSequence - Inherited Member Functions and Data 1477
IGSequenceAsDilTable 1479
IGSequenceAsDilTable - Hierarchy List . . . 1479
IGSequenceAsDilTable - Member Functions and Data by Group 1479
IGSequenceAsDilTable - Type Definitions. . . 1480
IGSequenceAsDilTable - Associated Globals 1481
IGSequenceAsDilTable - Inherited Member Functions and Data 1481
IGSequenceAsDilTableCursor. 1483
IGSequenceAsDilTableCursor - Hierarchy List 1483
IGSequenceAsDilTableCursor - Member Functions and Data by Group 1484
IGSequenceAsDilTableCursor - Inherited Member Functions and Data 1484
IGSequenceAsList 1485
IGSequenceAsList - Hierarchy List 1485
IGSequenceAsList - Member Functions and Data by Group 1485
IGSequenceAsList - Type Definitions 1486
IGSequenceAsList - Associated Globals . . . 1487
IGSequenceAsList - Inherited Member Functions and Data 1487
IGSequenceAsListCursor 1489
IGSequenceAsListCursor - Hierarchy List . . 1489
IGSequenceAsListCursor - Member Functions and Data by Group 1490
IGSequenceAsListCursor - Inherited Member Functions and Data 1490
IGSequenceAsTable 1491
IGSequenceAsTable - Hierarchy List 1491
IGSequenceAsTable - Member Functions and Data by Group 1491
IGSequenceAsTable - Type Definitions. . . . 1492
IGSequenceAsTable - Associated Globals . . . 1493
IGSequenceAsTable - Inherited Member Functions and Data 1493
IGSequenceAsTableCursor. 1495
IGSequenceAsTableCursor - Hierarchy List 1495
IGSequenceAsTableCursor - Member Functions and Data by Group 1496
IGSequenceAsTableCursor - Inherited Member Functions and Data 1496
IGSequenceCursor 1497
IGSequenceCursor - Hierarchy List. 1497

IGSequenceCursor - Member Functions and Data by Group 1497
IGSequenceCursor - Inherited Member Functions and Data 1498
IGSet 1499
IGSet - Hierarchy List 1499
IGSet - Member Functions and Data by Group 1499
IGSet - Type Definitions 1500
IGSet - Associated Globals. 1500
IGSet - Inherited Member Functions and Data 1501
IGSetAsAvlTree 1503
IGSetAsAvlTree - Hierarchy List. 1503
IGSetAsAvlTree - Member Functions and Data by Group 1503
IGSetAsAvlTree - Type Definitions 1504
IGSetAsAvlTree - Associated Globals 1504
IGSetAsAvlTree - Inherited Member Functions and Data 1505
IGSetAsAvlTreeCursor 1507
IGSetAsAvlTreeCursor - Hierarchy List . . . 1507
IGSetAsAvlTreeCursor - Member Functions and Data by Group 1507
IGSetAsAvlTreeCursor - Inherited Member Functions and Data 1507
IGSetAsBstTree 1508
IGSetAsBstTree - Hierarchy List 1508
IGSetAsBstTree - Member Functions and Data by Group 1509
IGSetAsBstTree - Type Definitions 1509
IGSetAsBstTree - Associated Globals 1510
IGSetAsBstTree - Inherited Member Functions and Data 1510
IGSetAsBstTreeCursor 1512
IGSetAsBstTreeCursor - Hierarchy List . . . 1512
IGSetAsBstTreeCursor - Member Functions and Data by Group 1512
IGSetAsBstTreeCursor - Inherited Member Functions and Data 1513
IGSetAsDilTable 1514
IGSetAsDilTable - Hierarchy List 1514
IGSetAsDilTable - Member Functions and Data by Group 1514
IGSetAsDilTable - Type Definitions. 1515
IGSetAsDilTable - Associated Globals 1515
IGSetAsDilTable - Inherited Member Functions and Data 1516
IGSetAsDilTableCursor. 1517
IGSetAsDilTableCursor - Hierarchy List . . . 1518
IGSetAsDilTableCursor - Member Functions and Data by Group 1518
IGSetAsDilTableCursor - Inherited Member Functions and Data 1518
IGSetAsHshTable. 1519
IGSetAsHshTable - Hierarchy List 1519
IGSetAsHshTable - Member Functions and Data by Group 1519
IGSetAsHshTable - Type Definitions 1520
IGSetAsHshTable - Associated Globals . . . 1521
IGSetAsHshTable - Inherited Member Functions and Data 1521
IGSetAsHshTableCursor 1523
IGSetAsHshTableCursor - Hierarchy List . . . 1523
IGSetAsHshTableCursor - Member Functions and Data by Group 1523
IGSetAsHshTableCursor - Inherited Member Functions and Data 1524
IGSetAsList 1524
IGSetAsList - Hierarchy List 1524
IGSetAsList - Member Functions and Data by Group 1525
IGSetAsList - Type Definitions 1525
IGSetAsList - Associated Globals 1526
IGSetAsList - Inherited Member Functions and Data 1526
IGSetAsListCursor 1528
IGSetAsListCursor - Hierarchy List. 1528
IGSetAsListCursor - Member Functions and Data by Group 1529
IGSetAsListCursor - Inherited Member Functions and Data 1529
IGSetAsTable 1530
IGSetAsTable - Hierarchy List 1530
IGSetAsTable - Member Functions and Data by Group 1530
IGSetAsTable - Type Definitions. 1531
IGSetAsTable - Associated Globals 1531
IGSetAsTable - Inherited Member Functions and Data 1532
IGSetAsTableCursor. 1534
IGSetAsTableCursor - Hierarchy List 1534
IGSetAsTableCursor - Member Functions and Data by Group 1534
IGSetAsTableCursor - Inherited Member Functions and Data 1534
IGSetCursor 1535
IGSetCursor - Hierarchy List 1535
IGSetCursor - Member Functions and Data by Group 1535
IGSetCursor - Inherited Member Functions and Data 1536
IGSortedBag 1537
IGSortedBag - Hierarchy List. 1537
IGSortedBag - Member Functions and Data by Group 1537
IGSortedBag - Type Definitions 1538
IGSortedBag - Associated Globals 1538
IGSortedBag - Inherited Member Functions and Data 1539
IGSortedBagAsAvlTree 1541
IGSortedBagAsAvlTree - Hierarchy List . . . 1541
IGSortedBagAsAvlTree - Member Functions and Data by Group 1541
IGSortedBagAsAvlTree - Type Definitions . . 1542
IGSortedBagAsAvlTree - Associated Globals 1543
IGSortedBagAsAvlTree - Inherited Member Functions and Data 1543
IGSortedBagAsAvlTreeCursor 1545
IGSortedBagAsAvlTreeCursor - Hierarchy List 1545
IGSortedBagAsAvlTreeCursor - Member Functions and Data by Group 1545
IGSortedBagAsAvlTreeCursor - Inherited Member Functions and Data 1546

IGSortedBagAsBstTree 1547
IGSortedBagAsBstTree - Hierarchy List . . . 1547
IGSortedBagAsBstTree - Member Functions and Data by Group 1547
IGSortedBagAsBstTree - Type Definitions. . . 1548
IGSortedBagAsBstTree - Associated Globals 1549
IGSortedBagAsBstTree - Inherited Member Functions and Data 1549
IGSortedBagAsBstTreeCursor. 1551
IGSortedBagAsBstTreeCursor - Hierarchy List 1551
IGSortedBagAsBstTreeCursor - Member Functions and Data by Group 1552
IGSortedBagAsBstTreeCursor - Inherited Member Functions and Data 1552
IGSortedBagAsDilTable. 1553
IGSortedBagAsDilTable - Hierarchy List . . . 1553
IGSortedBagAsDilTable - Member Functions and Data by Group 1554
IGSortedBagAsDilTable - Type Definitions . . 1554
IGSortedBagAsDilTable - Associated Globals 1555
IGSortedBagAsDilTable - Inherited Member Functions and Data 1555
IGSortedBagAsDilTableCursor 1558
IGSortedBagAsDilTableCursor - Hierarchy List 1558
IGSortedBagAsDilTableCursor - Member Functions and Data by Group 1558
IGSortedBagAsDilTableCursor - Inherited Member Functions and Data 1558
IGSortedBagAsList 1559
IGSortedBagAsList - Hierarchy List 1560
IGSortedBagAsList - Member Functions and Data by Group 1560
IGSortedBagAsList - Type Definitions 1561
IGSortedBagAsList - Associated Globals . . . 1561
IGSortedBagAsList - Inherited Member Functions and Data 1562
IGSortedBagAsListCursor 1564
IGSortedBagAsListCursor - Hierarchy List 1564
IGSortedBagAsListCursor - Member Functions and Data by Group 1564
IGSortedBagAsListCursor - Inherited Member Functions and Data 1565
IGSortedBagAsTable. 1566
IGSortedBagAsTable - Hierarchy List 1566
IGSortedBagAsTable - Member Functions and Data by Group 1566
IGSortedBagAsTable - Type Definitions . . . 1567
IGSortedBagAsTable - Associated Globals . . 1567
IGSortedBagAsTable - Inherited Member Functions and Data 1568
IGSortedBagAsTableCursor 1570
IGSortedBagAsTableCursor - Hierarchy List 1570
IGSortedBagAsTableCursor - Member Functions and Data by Group 1570
IGSortedBagAsTableCursor - Inherited Member Functions and Data 1571
IGSortedBagCursor 1572
IGSortedBagCursor - Hierarchy List 1572
IGSortedBagCursor - Member Functions and Data by Group 1572
IGSortedBagCursor - Inherited Member Functions and Data 1573
IGSortedMap 1574
IGSortedMap - Hierarchy List 1574
IGSortedMap - Member Functions and Data by Group 1574
IGSortedMap - Type Definitions. 1575
IGSortedMap - Associated Globals 1575
IGSortedMap - Inherited Member Functions and Data 1576
IGSortedMapAsAvlTree 1578
IGSortedMapAsAvlTree - Hierarchy List . . . 1578
IGSortedMapAsAvlTree - Member Functions and Data by Group 1578
IGSortedMapAsAvlTree - Type Definitions 1579
IGSortedMapAsAvlTree - Associated Globals 1580
IGSortedMapAsAvlTree - Inherited Member Functions and Data 1580
IGSortedMapAsAvlTreeCursor 1582
IGSortedMapAsAvlTreeCursor - Hierarchy List 1582
IGSortedMapAsAvlTreeCursor - Member Functions and Data by Group 1583
IGSortedMapAsAvlTreeCursor - Inherited Member Functions and Data 1583
IGSortedMapAsBstTree. 1584
IGSortedMapAsBstTree - Hierarchy List . . . 1584
IGSortedMapAsBstTree - Member Functions and Data by Group 1584
IGSortedMapAsBstTree - Type Definitions . . 1585
IGSortedMapAsBstTree - Associated Globals 1586
IGSortedMapAsBstTree - Inherited Member Functions and Data 1586
IGSortedMapAsBstTreeCursor 1588
IGSortedMapAsBstTreeCursor - Hierarchy List 1589
IGSortedMapAsBstTreeCursor - Member Functions and Data by Group 1589
IGSortedMapAsBstTreeCursor - Inherited Member Functions and Data 1589
IGSortedMapAsDilTable 1590
IGSortedMapAsDilTable - Hierarchy List. . . 1590
IGSortedMapAsDilTable - Member Functions and Data by Group 1591
IGSortedMapAsDilTable - Type Definitions 1591
IGSortedMapAsDilTable - Associated Globals 1592
IGSortedMapAsDilTable - Inherited Member Functions and Data 1592
IGSortedMapAsDilTableCursor 1595
IGSortedMapAsDilTableCursor - Hierarchy List 1595
IGSortedMapAsDilTableCursor - Member Functions and Data by Group 1595
IGSortedMapAsDilTableCursor - Inherited Member Functions and Data 1595
IGSortedMapAsList 1596
IGSortedMapAsList - Hierarchy List 1597
IGSortedMapAsList - Member Functions and Data by Group 1597
IGSortedMapAsList - Type Definitions . . . 1598
IGSortedMapAsList - Associated Globals. . . 1598
IGSortedMapAsList - Inherited Member Functions and Data 1599
IGSortedMapAsListCursor 1601

IGSortedMapAsListCursor - Hierarchy List 1601
IGSortedMapAsListCursor - Member Functions and Data by Group 1601
IGSortedMapAsListCursor - Inherited Member Functions and Data 1602
IGSortedMapAsTable 1603
IGSortedMapAsTable - Hierarchy List 1603
IGSortedMapAsTable - Member Functions and Data by Group 1603
IGSortedMapAsTable - Type Definitions . . . 1604
IGSortedMapAsTable - Associated Globals 1604
IGSortedMapAsTable - Inherited Member Functions and Data 1605
IGSortedMapAsTableCursor 1607
IGSortedMapAsTableCursor - Hierarchy List 1607
IGSortedMapAsTableCursor - Member Functions and Data by Group 1607
IGSortedMapAsTableCursor - Inherited Member Functions and Data 1608
IGSortedMapCursor 1609
IGSortedMapCursor - Hierarchy List 1609
IGSortedMapCursor - Member Functions and Data by Group 1609
IGSortedMapCursor - Inherited Member Functions and Data 1610
IGSortedRelation 1611
IGSortedRelation - Hierarchy List 1611
IGSortedRelation - Member Functions and Data by Group 1611
IGSortedRelation - Type Definitions 1612
IGSortedRelation - Associated Globals. . . . 1612
IGSortedRelation - Inherited Member Functions and Data 1613
IGSortedRelationAsDilTable 1615
IGSortedRelationAsDilTable - Hierarchy List 1615
IGSortedRelationAsDilTable - Member Functions and Data by Group 1615
IGSortedRelationAsDilTable - Type Definitions 1616
IGSortedRelationAsDilTable - Associated Globals 1617
IGSortedRelationAsDilTable - Inherited Member Functions and Data 1617
IGSortedRelationAsDilTableCursor 1619
IGSortedRelationAsDilTableCursor - Hierarchy List 1619
IGSortedRelationAsDilTableCursor - Member Functions and Data by Group 1619
IGSortedRelationAsDilTableCursor - Inherited Member Functions and Data 1620
IGSortedRelationAsTable 1621
IGSortedRelationAsTable - Hierarchy List . . 1621
IGSortedRelationAsTable - Member Functions and Data by Group 1621
IGSortedRelationAsTable - Type Definitions 1622
IGSortedRelationAsTable - Associated Globals 1623
IGSortedRelationAsTable - Inherited Member Functions and Data 1623
IGSortedRelationAsTableCursor 1625
IGSortedRelationAsTableCursor - Hierarchy List 1625
IGSortedRelationAsTableCursor - Member Functions and Data by Group 1625
IGSortedRelationAsTableCursor - Inherited Member Functions and Data 1626
IGSortedRelationCursor 1627
IGSortedRelationCursor - Hierarchy List . . . 1627
IGSortedRelationCursor - Member Functions and Data by Group 1627
IGSortedRelationCursor - Inherited Member Functions and Data 1628
IGSortedSet 1629
IGSortedSet - Hierarchy List 1629
IGSortedSet - Member Functions and Data by Group 1629
IGSortedSet - Type Definitions 1630
IGSortedSet - Associated Globals 1631
IGSortedSet - Inherited Member Functions and Data 1631
IGSortedSetAsAvlTree 1633
IGSortedSetAsAvlTree - Hierarchy List . . . 1633
IGSortedSetAsAvlTree - Member Functions and Data by Group 1633
IGSortedSetAsAvlTree - Type Definitions . . . 1634
IGSortedSetAsAvlTree - Associated Globals 1635
IGSortedSetAsAvlTree - Inherited Member Functions and Data 1635
IGSortedSetAsAvlTreeCursor 1637
IGSortedSetAsAvlTreeCursor - Hierarchy List 1637
IGSortedSetAsAvlTreeCursor - Member Functions and Data by Group 1638
IGSortedSetAsAvlTreeCursor - Inherited Member Functions and Data 1638
IGSortedSetAsBstTree 1639
IGSortedSetAsBstTree - Hierarchy List. . . . 1639
IGSortedSetAsBstTree - Member Functions and Data by Group 1639
IGSortedSetAsBstTree - Type Definitions . . . 1640
IGSortedSetAsBstTree - Associated Globals 1641
IGSortedSetAsBstTree - Inherited Member Functions and Data 1641
IGSortedSetAsBstTreeCursor 1643
IGSortedSetAsBstTreeCursor - Hierarchy List 1643
IGSortedSetAsBstTreeCursor - Member Functions and Data by Group 1644
IGSortedSetAsBstTreeCursor - Inherited Member Functions and Data 1644
IGSortedSetAsDilTable 1645
IGSortedSetAsDilTable - Hierarchy List . . . 1645
IGSortedSetAsDilTable - Member Functions and Data by Group 1645
IGSortedSetAsDilTable - Type Definitions . . 1646
IGSortedSetAsDilTable - Associated Globals 1647
IGSortedSetAsDilTable - Inherited Member Functions and Data 1647
IGSortedSetAsDilTableCursor 1649
IGSortedSetAsDilTableCursor - Hierarchy List 1649
IGSortedSetAsDilTableCursor - Member Functions and Data by Group 1650
IGSortedSetAsDilTableCursor - Inherited Member Functions and Data 1650
IGSortedSetAsList 1651

IGSortedSetAsList - Hierarchy List 1651
IGSortedSetAsList - Member Functions and Data by Group 1652
IGSortedSetAsList - Type Definitions 1652
IGSortedSetAsList - Associated Globals . . . 1653
IGSortedSetAsList - Inherited Member Functions and Data 1653
IGSortedSetAsListCursor 1655
IGSortedSetAsListCursor - Hierarchy List . . 1656
IGSortedSetAsListCursor - Member Functions and Data by Group 1656
IGSortedSetAsListCursor - Inherited Member Functions and Data 1656
IGSortedSetAsTable 1657
IGSortedSetAsTable - Hierarchy List 1657
IGSortedSetAsTable - Member Functions and Data by Group 1658
IGSortedSetAsTable - Type Definitions . . . 1658
IGSortedSetAsTable - Associated Globals . . . 1659
IGSortedSetAsTable - Inherited Member Functions and Data 1659
IGSortedSetAsTableCursor 1661
IGSortedSetAsTableCursor - Hierarchy List 1662
IGSortedSetAsTableCursor - Member Functions and Data by Group 1662
IGSortedSetAsTableCursor - Inherited Member Functions and Data 1662
IGSortedSetCursor 1663
IGSortedSetCursor - Hierarchy List. 1663
IGSortedSetCursor - Member Functions and Data by Group 1663
IGSortedSetCursor - Inherited Member Functions and Data 1664
IGStack 1665
IGStack - Hierarchy List 1665
IGStack - Member Functions and Data by Group 1665
IGStack - Type Definitions. 1666
IGStack - Associated Globals 1667
IGStack - Inherited Member Functions and Data 1667
IGStackAsDilTable 1669
IGStackAsDilTable - Hierarchy List. 1669
IGStackAsDilTable - Member Functions and Data by Group 1669
IGStackAsDilTable - Type Definitions 1670
IGStackAsDilTable - Associated Globals . . . 1670
IGStackAsDilTable - Inherited Member Functions and Data 1671
IGStackAsDilTableCursor 1672
IGStackAsDilTableCursor - Hierarchy List . . 1672
IGStackAsDilTableCursor - Member Functions and Data by Group 1673
IGStackAsDilTableCursor - Inherited Member Functions and Data 1673
IGStackAsList 1674
IGStackAsList - Hierarchy List 1674
IGStackAsList - Member Functions and Data by Group 1674
IGStackAsList - Type Definitions 1675
IGStackAsList - Associated Globals. 1676
IGStackAsList - Inherited Member Functions and Data 1676
IGStackAsListCursor 1678
IGStackAsListCursor - Hierarchy List 1678
IGStackAsListCursor - Member Functions and Data by Group 1678
IGStackAsListCursor - Inherited Member Functions and Data 1678
IGStackAsTable 1679
IGStackAsTable - Hierarchy List. 1679
IGStackAsTable - Member Functions and Data by Group 1680
IGStackAsTable - Type Definitions 1680
IGStackAsTable - Associated Globals 1681
IGStackAsTable - Inherited Member Functions and Data 1681
IGStackAsTableCursor 1683
IGStackAsTableCursor - Hierarchy List . . . 1683
IGStackAsTableCursor - Member Functions and Data by Group 1683
IGStackAsTableCursor - Inherited Member Functions and Data 1684
IGStackCursor. 1685
IGStackCursor - Hierarchy List 1685
IGStackCursor - Member Functions and Data by Group 1685
IGStackCursor - Inherited Member Functions and Data 1685
IGTree 1687
IGTree - Hierarchy List. 1687
IGTree - Member Functions and Data by Group 1687
IGTree - Inherited Member Functions and Data 1687
IHeap 1689
IHeap - Hierarchy List 1689
IHeap - Member Functions and Data by Group 1689
IHeap - Type Definitions 1689
IHeap - Associated Globals 1690
IHeap - Inherited Member Functions and Data 1690
IHeapAsDilTable 1691
IHeapAsDilTable - Hierarchy List 1691
IHeapAsDilTable - Member Functions and Data by Group 1691
IHeapAsDilTable - Type Definitions 1691
IHeapAsDilTable - Associated Globals. . . . 1692
IHeapAsDilTable - Inherited Member Functions and Data 1692
IHeapAsList 1692
IHeapAsList - Hierarchy List. 1693
IHeapAsList - Member Functions and Data by Group 1693
IHeapAsList - Type Definitions 1693
IHeapAsList - Associated Globals 1693
IHeapAsList - Inherited Member Functions and Data 1694
IHeapAsTable 1694
IHeapAsTable - Hierarchy List 1694
IHeapAsTable - Member Functions and Data by Group 1694
IHeapAsTable - Type Definitions 1695
IHeapAsTable - Associated Globals. 1695

IHeapAsTable - Inherited Member Functions and Data 1695
IInternalApplicator 1696
IInternalApplicator - Hierarchy List 1696
IInternalApplicator - Member Functions and Data by Group 1696
IInternalApplicator - Inherited Member Functions and Data 1696
IKeyBag 1696
IKeyBag - Hierarchy List 1697
IKeyBag - Member Functions and Data by Group 1697
IKeyBag - Type Definitions 1697
IKeyBag - Associated Globals. 1698
IKeyBag - Inherited Member Functions and Data 1698
IKeyBagAsDilTable 1700
IKeyBagAsDilTable - Hierarchy List 1700
IKeyBagAsDilTable - Member Functions and Data by Group 1700
IKeyBagAsDilTable - Type Definitions 1701
IKeyBagAsDilTable - Associated Globals . . . 1701
IKeyBagAsDilTable - Inherited Member Functions and Data 1701
IKeyBagAsHshTable. 1703
IKeyBagAsHshTable - Hierarchy List 1704
IKeyBagAsHshTable - Member Functions and Data by Group 1704
IKeyBagAsHshTable - Type Definitions . . . 1704
IKeyBagAsHshTable - Associated Globals . . 1704
IKeyBagAsHshTable - Inherited Member Functions and Data 1705
IKeyBagAsList 1707
IKeyBagAsList - Hierarchy List 1707
IKeyBagAsList - Member Functions and Data by Group 1707
IKeyBagAsList - Type Definitions 1708
IKeyBagAsList - Associated Globals 1708
IKeyBagAsList - Inherited Member Functions and Data 1708
IKeyBagAsTable 1710
IKeyBagAsTable - Hierarchy List 1710
IKeyBagAsTable - Member Functions and Data by Group 1711
IKeyBagAsTable - Type Definitions 1711
IKeyBagAsTable - Associated Globals 1711
IKeyBagAsTable - Inherited Member Functions and Data 1711
IKeySet 1713
IKeySet - Hierarchy List 1714
IKeySet - Member Functions and Data by Group 1714
IKeySet - Type Definitions. 1714
IKeySet - Associated Globals 1715
IKeySet - Inherited Member Functions and Data 1715
IKeySetAsAvlTree 1717
IKeySetAsAvlTree - Hierarchy List 1717
IKeySetAsAvlTree - Member Functions and Data by Group 1718
IKeySetAsAvlTree - Type Definitions 1718
IKeySetAsAvlTree - Associated Globals . . . 1718
IKeySetAsAvlTree - Inherited Member Functions and Data 1719
IKeySetAsBstTree. 1720
IKeySetAsBstTree - Hierarchy List 1721
IKeySetAsBstTree - Member Functions and Data by Group 1721
IKeySetAsBstTree - Type Definitions 1721
IKeySetAsBstTree - Associated Globals . . . 1722
IKeySetAsBstTree - Inherited Member Functions and Data 1722
IKeySetAsDilTable 1724
IKeySetAsDilTable - Hierarchy List. 1724
IKeySetAsDilTable - Member Functions and Data by Group 1724
IKeySetAsDilTable - Type Definitions 1725
IKeySetAsDilTable - Associated Globals . . . 1725
IKeySetAsDilTable - Inherited Member Functions and Data 1725
IKeySetAsHshTable 1727
IKeySetAsHshTable - Hierarchy List 1728
IKeySetAsHshTable - Member Functions and Data by Group 1728
IKeySetAsHshTable - Type Definitions. . . . 1728
IKeySetAsHshTable - Associated Globals . . . 1729
IKeySetAsHshTable - Inherited Member Functions and Data 1729
IKeySetAsList 1731
IKeySetAsList - Hierarchy List 1731
IKeySetAsList - Member Functions and Data by Group 1731
IKeySetAsList - Type Definitions 1732
IKeySetAsList - Associated Globals. 1732
IKeySetAsList - Inherited Member Functions and Data 1732
IKeySetAsTable 1734
IKeySetAsTable - Hierarchy List. 1735
IKeySetAsTable - Member Functions and Data by Group 1735
IKeySetAsTable - Type Definitions 1735
IKeySetAsTable - Associated Globals 1735
IKeySetAsTable - Inherited Member Functions and Data 1736
IKeySortedBag 1738
IKeySortedBag - Hierarchy List 1738
IKeySortedBag - Member Functions and Data by Group 1738
IKeySortedBag - Type Definitions 1739
IKeySortedBag - Associated Globals 1739
IKeySortedBag - Inherited Member Functions and Data 1739
IKeySortedBagAsDilTable 1741
IKeySortedBagAsDilTable - Hierarchy List . . 1742
IKeySortedBagAsDilTable - Member Functions and Data by Group 1742
IKeySortedBagAsDilTable - Type Definitions 1742
IKeySortedBagAsDilTable - Associated Globals 1742
IKeySortedBagAsDilTable - Inherited Member Functions and Data 1743
IKeySortedBagAsList 1745
IKeySortedBagAsList - Hierarchy List 1745

IKeySortedBagAsList - Member Functions and Data by Group 1746
IKeySortedBagAsList - Type Definitions . . . 1746
IKeySortedBagAsList - Associated Globals . . 1746
IKeySortedBagAsList - Inherited Member Functions and Data 1746
IKeySortedBagAsTable 1749
IKeySortedBagAsTable - Hierarchy List . . . 1749
IKeySortedBagAsTable - Member Functions and Data by Group 1749
IKeySortedBagAsTable - Type Definitions . . 1750
IKeySortedBagAsTable - Associated Globals 1750
IKeySortedBagAsTable - Inherited Member Functions and Data 1750
IKeySortedSet 1752
IKeySortedSet - Hierarchy List 1753
IKeySortedSet - Member Functions and Data by Group 1753
IKeySortedSet - Type Definitions 1753
IKeySortedSet - Associated Globals. 1754
IKeySortedSet - Inherited Member Functions and Data 1754
IKeySortedSetAsAvlTree 1756
IKeySortedSetAsAvlTree - Hierarchy List . . . 1757
IKeySortedSetAsAvlTree - Member Functions and Data by Group 1757
IKeySortedSetAsAvlTree - Type Definitions 1757
IKeySortedSetAsAvlTree - Associated Globals 1757
IKeySortedSetAsAvlTree - Inherited Member Functions and Data 1758
IKeySortedSetAsBstTree 1760
IKeySortedSetAsBstTree - Hierarchy List . . . 1760
IKeySortedSetAsBstTree - Member Functions and Data by Group 1761
IKeySortedSetAsBstTree - Type Definitions 1761
IKeySortedSetAsBstTree - Associated Globals 1761
IKeySortedSetAsBstTree - Inherited Member Functions and Data 1762
IKeySortedSetAsDilTable 1764
IKeySortedSetAsDilTable - Hierarchy List . . 1764
IKeySortedSetAsDilTable - Member Functions and Data by Group 1765
IKeySortedSetAsDilTable - Type Definitions 1765
IKeySortedSetAsDilTable - Associated Globals 1765
IKeySortedSetAsDilTable - Inherited Member Functions and Data 1766
IKeySortedSetAsList. 1768
IKeySortedSetAsList - Hierarchy List 1768
IKeySortedSetAsList - Member Functions and Data by Group 1768
IKeySortedSetAsList - Type Definitions . . . 1769
IKeySortedSetAsList - Associated Globals . . 1769
IKeySortedSetAsList - Inherited Member Functions and Data 1769
IKeySortedSetAsTable 1772
IKeySortedSetAsTable - Hierarchy List . . . 1772
IKeySortedSetAsTable - Member Functions and Data by Group 1772
IKeySortedSetAsTable - Type Definitions . . . 1773
IKeySortedSetAsTable - Associated Globals 1773
IKeySortedSetAsTable - Inherited Member Functions and Data 1773
IMap 1775
IMap - Hierarchy List 1776
IMap - Member Functions and Data by Group 1776
IMap - Type Definitions 1776
IMap - Associated Globals. 1777
IMap - Inherited Member Functions and Data 1777
IMapAsAvlTree 1779
IMapAsAvlTree - Hierarchy List. 1779
IMapAsAvlTree - Member Functions and Data by Group 1780
IMapAsAvlTree - Type Definitions 1780
IMapAsAvlTree - Associated Globals 1780
IMapAsAvlTree - Inherited Member Functions and Data 1781
IMapAsBstTree 1783
IMapAsBstTree - Hierarchy List 1783
IMapAsBstTree - Member Functions and Data by Group 1784
IMapAsBstTree - Type Definitions 1784
IMapAsBstTree - Associated Globals 1784
IMapAsBstTree - Inherited Member Functions and Data 1784
IMapAsDilTable 1787
IMapAsDilTable - Hierarchy List 1787
IMapAsDilTable - Member Functions and Data by Group 1787
IMapAsDilTable - Type Definitions. 1788
IMapAsDilTable - Associated Globals 1788
IMapAsDilTable - Inherited Member Functions and Data 1788
IMapAsHshTable. 1791
IMapAsHshTable - Hierarchy List 1791
IMapAsHshTable - Member Functions and Data by Group 1791
IMapAsHshTable - Type Definitions 1792
IMapAsHshTable - Associated Globals. . . . 1792
IMapAsHshTable - Inherited Member Functions and Data 1792
IMapAsList. 1795
IMapAsList - Hierarchy List 1795
IMapAsList - Member Functions and Data by Group 1795
IMapAsList - Type Definitions 1796
IMapAsList - Associated Globals 1796
IMapAsList - Inherited Member Functions and Data 1796
IMapAsTable 1798
IMapAsTable - Hierarchy List 1799
IMapAsTable - Member Functions and Data by Group 1799
IMapAsTable - Type Definitions. 1799
IMapAsTable - Associated Globals 1800
IMapAsTable - Inherited Member Functions and Data 1800
IMngElemPointer. 1802
IMngElemPointer - Hierarchy List 1802
IMngElemPointer - Member Functions and Data by Group 1802

IMngElemPointer - Inherited Member Functions and Data 1803
IMngPointer 1804
IMngPointer - Hierarchy List 1804
IMngPointer - Member Functions and Data by Group 1804
IMngPointer - Inherited Member Functions and Data 1805
IMultiwayTree. 1805
IMultiwayTree - Hierarchy List 1805
IMultiwayTree - Member Functions and Data by Group 1806
IMultiwayTree - Type Definitions 1806
IMultiwayTree - Associated Globals 1806
IMultiwayTree - Inherited Member Functions and Data 1807
IOrderedCursor 1808
IOrderedCursor - Hierarchy List. 1809
IOrderedCursor - Member Functions and Data by Group 1810
IOrderedCursor - Inherited Member Functions and Data 1811
IPriorityQueue 1812
IPriorityQueue - Hierarchy List 1812
IPriorityQueue - Member Functions and Data by Group 1812
IPriorityQueue - Type Definitions 1812
IPriorityQueue - Associated Globals 1813
IPriorityQueue - Inherited Member Functions and Data 1813
IPriorityQueueAsDilTable 1815
IPriorityQueueAsDilTable - Hierarchy List 1815
IPriorityQueueAsDilTable - Member Functions and Data by Group 1815
IPriorityQueueAsDilTable - Type Definitions 1816
IPriorityQueueAsDilTable - Associated Globals 1816
IPriorityQueueAsDilTable - Inherited Member Functions and Data 1816
IPriorityQueueAsList 1818
IPriorityQueueAsList - Hierarchy List 1819
IPriorityQueueAsList - Member Functions and Data by Group 1819
IPriorityQueueAsList - Type Definitions . . . 1819
IPriorityQueueAsList - Associated Globals 1819
IPriorityQueueAsList - Inherited Member Functions and Data 1820
IPriorityQueueAsTable 1821
IPriorityQueueAsTable - Hierarchy List . . . 1822
IPriorityQueueAsTable - Member Functions and Data by Group 1822
IPriorityQueueAsTable - Type Definitions . . 1822
IPriorityQueueAsTable - Associated Globals 1823
IPriorityQueueAsTable - Inherited Member Functions and Data 1823
IQueue 1825
IQueue - Hierarchy List 1825
IQueue - Member Functions and Data by Group 1825
IQueue - Type Definitions 1826
IQueue - Associated Globals 1826
IQueue - Inherited Member Functions and Data 1826
IQueueAsDilTable 1828
IQueueAsDilTable - Hierarchy List 1828
IQueueAsDilTable - Member Functions and Data by Group 1828
IQueueAsDilTable - Type Definitions 1829
IQueueAsDilTable - Associated Globals . . . 1829
IQueueAsDilTable - Inherited Member Functions and Data 1829
IQueueAsList 1831
IQueueAsList - Hierarchy List 1831
IQueueAsList - Member Functions and Data by Group 1831
IQueueAsList - Type Definitions. 1832
IQueueAsList - Associated Globals 1832
IQueueAsList - Inherited Member Functions and Data 1832
IQueueAsTable 1834
IQueueAsTable - Hierarchy List 1834
IQueueAsTable - Member Functions and Data by Group 1835
IQueueAsTable - Type Definitions 1835
IQueueAsTable - Associated Globals 1835
IQueueAsTable - Inherited Member Functions and Data 1836
IRBag 1837
IRBag - Hierarchy List 1837
IRBag - Member Functions and Data by Group 1838
IRBag - Type Definitions 1838
IRBag - Inherited Member Functions and Data 1838
IRDeque. 1840
IRDeque - Hierarchy List 1840
IRDeque - Member Functions and Data by Group 1840
IRDeque - Type Definitions 1840
IRDeque - Inherited Member Functions and Data 1841
IRelation 1842
IRelation - Hierarchy List 1842
IRelation - Member Functions and Data by Group 1843
IRelation - Type Definitions 1843
IRelation - Associated Globals 1843
IRelation - Inherited Member Functions and Data 1843
IRelationAsDilTable 1846
IRelationAsDilTable - Hierarchy List 1846
IRelationAsDilTable - Member Functions and Data by Group 1846
IRelationAsDilTable - Type Definitions . . . 1846
IRelationAsDilTable - Associated Globals . . . 1847
IRelationAsDilTable - Inherited Member Functions and Data 1847
IRelationAsHshTable 1849
IRelationAsHshTable - Hierarchy List 1849
IRelationAsHshTable - Member Functions and Data by Group 1850
IRelationAsHshTable - Type Definitions . . . 1850
IRelationAsHshTable - Associated Globals . . 1850

IRelationAsHshTable - Inherited Member Functions and Data 1850
IRelationAsList 1853
IRelationAsList - Hierarchy List 1853
IRelationAsList - Member Functions and Data by Group 1853
IRelationAsList - Type Definitions 1853
IRelationAsList - Associated Globals 1854
IRelationAsList - Inherited Member Functions and Data 1854
IRelationAsTable 1856
IRelationAsTable - Hierarchy List 1856
IRelationAsTable - Member Functions and Data by Group 1857
IRelationAsTable - Type Definitions 1857
IRelationAsTable - Associated Globals 1857
IRelationAsTable - Inherited Member Functions and Data 1857
IREqualitySequence 1860
IREqualitySequence - Hierarchy List 1860
IREqualitySequence - Member Functions and Data by Group 1860
IREqualitySequence - Type Definitions . . . 1860
IREqualitySequence - Inherited Member Functions and Data 1861
IRestrictedAccessCollectionGuard 1862
IRestrictedAccessCollectionGuard - Hierarchy List 1862
IRestrictedAccessCollectionGuard - Member Functions and Data by Group 1863
IRestrictedAccessCollectionGuard - Inherited Member Functions and Data 1863
IRHeap 1863
IRHeap - Hierarchy List 1863
IRHeap - Member Functions and Data by Group 1864
IRHeap - Type Definitions. 1864
IRHeap - Inherited Member Functions and Data 1864
IRKeyBag 1864
IRKeyBag - Hierarchy List. 1864
IRKeyBag - Member Functions and Data by Group 1865
IRKeyBag - Type Definitions 1865
IRKeyBag - Inherited Member Functions and Data 1865
IRKeySet 1867
IRKeySet - Hierarchy List 1867
IRKeySet - Member Functions and Data by Group 1867
IRKeySet - Type Definitions 1867
IRKeySet - Inherited Member Functions and Data 1867
IRKeySortedBag 1869
IRKeySortedBag - Hierarchy List 1869
IRKeySortedBag - Member Functions and Data by Group 1869
IRKeySortedBag - Type Definitions. 1870
IRKeySortedBag - Inherited Member Functions and Data 1870
IRKeySortedSet 1872
IRKeySortedSet - Hierarchy List. 1872
IRKeySortedSet - Member Functions and Data by Group 1872
IRKeySortedSet - Type Definitions 1872
IRKeySortedSet - Inherited Member Functions and Data 1873
IRMap 1874
IRMap - Hierarchy List. 1875
IRMap - Member Functions and Data by Group 1875
IRMap - Type Definitions 1875
IRMap - Inherited Member Functions and Data 1875
IRPriorityQueue 1877
IRPriorityQueue - Hierarchy List 1877
IRPriorityQueue - Member Functions and Data by Group 1878
IRPriorityQueue - Type Definitions. 1878
IRPriorityQueue - Inherited Member Functions and Data 1878
IRQueue 1880
IRQueue - Hierarchy List 1880
IRQueue - Member Functions and Data by Group 1880
IRQueue - Type Definitions 1880
IRQueue - Inherited Member Functions and Data 1880
IRRelation 1882
IRRelation - Hierarchy List 1882
IRRelation - Member Functions and Data by Group 1882
IRRelation - Type Definitions. 1882
IRRelation - Inherited Member Functions and Data 1883
IRSequence. 1884
IRSequence - Hierarchy List 1885
IRSequence - Member Functions and Data by Group 1885
IRSequence - Type Definitions 1885
IRSequence - Inherited Member Functions and Data 1885
IRSet 1887
IRSet - Hierarchy List 1887
IRSet - Member Functions and Data by Group 1888
IRSet - Type Definitions 1888
IRSet - Inherited Member Functions and Data 1888
IRSortedBag 1890
IRSortedBag - Hierarchy List 1890
IRSortedBag - Member Functions and Data by Group 1890
IRSortedBag - Type Definitions 1890
IRSortedBag - Inherited Member Functions and Data 1890
IRSortedMap 1892
IRSortedMap - Hierarchy List 1893
IRSortedMap - Member Functions and Data by Group 1893
IRSortedMap - Type Definitions 1893
IRSortedMap - Inherited Member Functions and Data 1893
IRSortedRelation 1895
IRSortedRelation - Hierarchy List 1895

IRSortedRelation - Member Functions and Data by Group 1896
IRSortedRelation - Type Definitions 1896
IRSortedRelation - Inherited Member Functions and Data 1896
IRSortedSet. 1898
IRSortedSet - Hierarchy List 1898
IRSortedSet - Member Functions and Data by Group 1898
IRSortedSet - Type Definitions 1899
IRSortedSet - Inherited Member Functions and Data 1899
IRStack 1901
IRStack - Hierarchy List 1901
IRStack - Member Functions and Data by Group 1901
IRStack - Type Definitions. 1901
IRStack - Inherited Member Functions and Data 1902
ISequence 1903
ISequence - Hierarchy List 1903
ISequence - Member Functions and Data by Group 1904
ISequence - Type Definitions 1904
ISequence - Associated Globals 1904
ISequence - Inherited Member Functions and Data 1904
ISequenceAsDilTable 1907
ISequenceAsDilTable - Hierarchy List 1907
ISequenceAsDilTable - Member Functions and Data by Group 1907
ISequenceAsDilTable - Type Definitions . . . 1908
ISequenceAsDilTable - Associated Globals . . 1908
ISequenceAsDilTable - Inherited Member Functions and Data 1908
ISequenceAsList 1911
ISequenceAsList - Hierarchy List 1911
ISequenceAsList - Member Functions and Data by Group 1911
ISequenceAsList - Type Definitions. 1912
ISequenceAsList - Associated Globals 1912
ISequenceAsList - Inherited Member Functions and Data 1912
ISequenceAsTable 1915
ISequenceAsTable - Hierarchy List 1915
ISequenceAsTable - Member Functions and Data by Group 1915
ISequenceAsTable - Type Definitions 1915
ISequenceAsTable - Associated Globals . . . 1916
ISequenceAsTable - Inherited Member Functions and Data 1916
ISet 1918
ISet - Hierarchy List. 1919
ISet - Member Functions and Data by Group 1919
ISet - Type Definitions 1919
ISet - Associated Globals 1919
ISet - Inherited Member Functions and Data 1920
ISetAsAvlTree 1922
ISetAsAvlTree - Hierarchy List 1922
ISetAsAvlTree - Member Functions and Data by Group 1922
ISetAsAvlTree - Type Definitions 1922
ISetAsAvlTree - Associated Globals. 1923
ISetAsAvlTree - Inherited Member Functions and Data 1923
ISetAsBstTree 1925
ISetAsBstTree - Hierarchy List 1925
ISetAsBstTree - Member Functions and Data by Group 1926
ISetAsBstTree - Type Definitions. 1926
ISetAsBstTree - Associated Globals 1926
ISetAsBstTree - Inherited Member Functions and Data 1926
ISetAsDilTable. 1929
ISetAsDilTable - Hierarchy List 1929
ISetAsDilTable - Member Functions and Data by Group 1929
ISetAsDilTable - Type Definitions 1929
ISetAsDilTable - Associated Globals 1930
ISetAsDilTable - Inherited Member Functions and Data 1930
ISetAsHshTable 1932
ISetAsHshTable - Hierarchy List. 1932
ISetAsHshTable - Member Functions and Data by Group 1932
ISetAsHshTable - Type Definitions 1933
ISetAsHshTable - Associated Globals 1933
ISetAsHshTable - Inherited Member Functions and Data 1933
ISetAsList 1935
ISetAsList - Hierarchy List 1936
ISetAsList - Member Functions and Data by Group 1936
ISetAsList - Type Definitions 1936
ISetAsList - Associated Globals 1936
ISetAsList - Inherited Member Functions and Data 1937
ISetAsTable. 1939
ISetAsTable - Hierarchy List 1939
ISetAsTable - Member Functions and Data by Group 1939
ISetAsTable - Type Definitions 1939
ISetAsTable - Associated Globals 1940
ISetAsTable - Inherited Member Functions and Data 1940
ISortedBag 1942
ISortedBag - Hierarchy List 1942
ISortedBag - Member Functions and Data by Group 1943
ISortedBag - Type Definitions 1943
ISortedBag - Associated Globals. 1943
ISortedBag - Inherited Member Functions and Data 1943
ISortedBagAsAvlTree 1946
ISortedBagAsAvlTree - Hierarchy List 1946
ISortedBagAsAvlTree - Member Functions and Data by Group 1946
ISortedBagAsAvlTree - Type Definitions . . . 1947
ISortedBagAsAvlTree - Associated Globals . . 1947
ISortedBagAsAvlTree - Inherited Member Functions and Data 1947
ISortedBagAsBstTree 1950

ISortedBagAsBstTree - Hierarchy List 1950
ISortedBagAsBstTree - Member Functions and Data by Group 1950
ISortedBagAsBstTree - Type Definitions . . . 1950
ISortedBagAsBstTree - Associated Globals . . 1951
ISortedBagAsBstTree - Inherited Member Functions and Data 1951
ISortedBagAsDilTable 1953
ISortedBagAsDilTable - Hierarchy List. . . . 1954
ISortedBagAsDilTable - Member Functions and Data by Group 1954
ISortedBagAsDilTable - Type Definitions . . . 1954
ISortedBagAsDilTable - Associated Globals 1954
ISortedBagAsDilTable - Inherited Member Functions and Data 1955
ISortedBagAsList. 1957
ISortedBagAsList - Hierarchy List 1957
ISortedBagAsList - Member Functions and Data by Group 1958
ISortedBagAsList - Type Definitions 1958
ISortedBagAsList - Associated Globals. . . . 1958
ISortedBagAsList - Inherited Member Functions and Data 1958
ISortedBagAsTable 1961
ISortedBagAsTable - Hierarchy List. 1961
ISortedBagAsTable - Member Functions and Data by Group 1961
ISortedBagAsTable - Type Definitions 1962
ISortedBagAsTable - Associated Globals . . . 1962
ISortedBagAsTable - Inherited Member Functions and Data 1962
ISortedMap 1965
ISortedMap - Hierarchy List 1965
ISortedMap - Member Functions and Data by Group 1965
ISortedMap - Type Definitions 1966
ISortedMap - Associated Globals 1966
ISortedMap - Inherited Member Functions and Data 1966
ISortedMapAsAvlTree 1968
ISortedMapAsAvlTree - Hierarchy List . . . 1969
ISortedMapAsAvlTree - Member Functions and Data by Group 1969
ISortedMapAsAvlTree - Type Definitions . . . 1969
ISortedMapAsAvlTree - Associated Globals 1970
ISortedMapAsAvlTree - Inherited Member Functions and Data 1970
ISortedMapAsBstTree 1972
ISortedMapAsBstTree - Hierarchy List. . . . 1973
ISortedMapAsBstTree - Member Functions and Data by Group 1973
ISortedMapAsBstTree - Type Definitions . . . 1973
ISortedMapAsBstTree - Associated Globals 1974
ISortedMapAsBstTree - Inherited Member Functions and Data 1974
ISortedMapAsDilTable 1976
ISortedMapAsDilTable - Hierarchy List . . . 1977
ISortedMapAsDilTable - Member Functions and Data by Group 1977
ISortedMapAsDilTable - Type Definitions. . . 1977
ISortedMapAsDilTable - Associated Globals 1977
ISortedMapAsDilTable - Inherited Member Functions and Data 1978
ISortedMapAsList 1980
ISortedMapAsList - Hierarchy List 1980
ISortedMapAsList - Member Functions and Data by Group 1981
ISortedMapAsList - Type Definitions 1981
ISortedMapAsList - Associated Globals . . . 1981
ISortedMapAsList - Inherited Member Functions and Data 1982
ISortedMapAsTable 1984
ISortedMapAsTable - Hierarchy List 1984
ISortedMapAsTable - Member Functions and Data by Group 1985
ISortedMapAsTable - Type Definitions. . . . 1985
ISortedMapAsTable - Associated Globals . . . 1985
ISortedMapAsTable - Inherited Member Functions and Data 1985
ISortedRelation 1988
ISortedRelation - Hierarchy List 1988
ISortedRelation - Member Functions and Data by Group 1988
ISortedRelation - Type Definitions 1989
ISortedRelation - Associated Globals 1989
ISortedRelation - Inherited Member Functions and Data 1989
ISortedRelationAsDilTable. 1992
ISortedRelationAsDilTable - Hierarchy List 1992
ISortedRelationAsDilTable - Member Functions and Data by Group 1992
ISortedRelationAsDilTable - Type Definitions 1993
ISortedRelationAsDilTable - Associated Globals 1993
ISortedRelationAsDilTable - Inherited Member Functions and Data 1993
ISortedRelationAsTable. 1995
ISortedRelationAsTable - Hierarchy List . . . 1996
ISortedRelationAsTable - Member Functions and Data by Group 1996
ISortedRelationAsTable - Type Definitions . . 1996
ISortedRelationAsTable - Associated Globals 1997
ISortedRelationAsTable - Inherited Member Functions and Data 1997
ISortedSet 1999
ISortedSet - Hierarchy List 2000
ISortedSet - Member Functions and Data by Group 2000
ISortedSet - Type Definitions 2000
ISortedSet - Associated Globals 2001
ISortedSet - Inherited Member Functions and Data 2001
ISortedSetAsAvlTree. 2003
ISortedSetAsAvlTree - Hierarchy List 2003
ISortedSetAsAvlTree - Member Functions and Data by Group 2004
ISortedSetAsAvlTree - Type Definitions . . . 2004
ISortedSetAsAvlTree - Associated Globals . . 2004
ISortedSetAsAvlTree - Inherited Member Functions and Data 2005
ISortedSetAsBstTree 2007
ISortedSetAsBstTree - Hierarchy List 2007

ISortedSetAsBstTree - Member Functions and Data by Group 2007
ISortedSetAsBstTree - Type Definitions . . . 2008
ISortedSetAsBstTree - Associated Globals . . . 2008
ISortedSetAsBstTree - Inherited Member Functions and Data 2008
ISortedSetAsDilTable 2011
ISortedSetAsDilTable - Hierarchy List 2011
ISortedSetAsDilTable - Member Functions and Data by Group 2011
ISortedSetAsDilTable - Type Definitions . . . 2011
ISortedSetAsDilTable - Associated Globals . . 2012
ISortedSetAsDilTable - Inherited Member Functions and Data 2012
ISortedSetAsList 2014
ISortedSetAsList - Hierarchy List 2015
ISortedSetAsList - Member Functions and Data by Group 2015
ISortedSetAsList - Type Definitions 2015
ISortedSetAsList - Associated Globals 2015
ISortedSetAsList - Inherited Member Functions and Data 2016
ISortedSetAsTable 2018
ISortedSetAsTable - Hierarchy List 2018
ISortedSetAsTable - Member Functions and Data by Group 2019
ISortedSetAsTable - Type Definitions 2019
ISortedSetAsTable - Associated Globals . . . 2019
ISortedSetAsTable - Inherited Member Functions and Data 2019
IStack 2022
IStack - Hierarchy List 2022
IStack - Member Functions and Data by Group 2022
IStack - Type Definitions 2023
IStack - Associated Globals 2023
IStack - Inherited Member Functions and Data 2023
IStackAsDilTable 2025
IStackAsDilTable - Hierarchy List 2025
IStackAsDilTable - Member Functions and Data by Group 2025
IStackAsDilTable - Type Definitions 2026
IStackAsDilTable - Associated Globals 2026
IStackAsDilTable - Inherited Member Functions and Data 2026
IStackAsList 2028
IStackAsList - Hierarchy List 2028
IStackAsList - Member Functions and Data by Group 2028
IStackAsList - Type Definitions 2029
IStackAsList - Associated Globals 2029
IStackAsList - Inherited Member Functions and Data 2029
IStackAsTable 2031
IStackAsTable - Hierarchy List 2031
IStackAsTable - Member Functions and Data by Group 2032
IStackAsTable - Type Definitions 2032
IStackAsTable - Associated Globals 2032
IStackAsTable - Inherited Member Functions and Data 2033
IStreamablePointer 2034
IStreamablePointer - Hierarchy List. 2034
IStreamablePointer - Member Functions and Data by Group 2035
IStreamablePointer - Inherited Member Functions and Data 2036
ITree 2036
ITree - Hierarchy List 2036
ITree - Member Functions and Data by Group 2036
ITree - Inherited Member Functions and Data 2036
ITreeCollectionGuard 2038
ITreeCollectionGuard - Hierarchy List 2038
ITreeCollectionGuard - Member Functions and Data by Group 2038
ITreeCollectionGuard - Inherited Member Functions and Data 2039
ITreeCursor 2039
ITreeCursor - Hierarchy List 2039
ITreeCursor - Member Functions and Data by Group 2039
ITreeCursor - Inherited Member Functions and Data 2043
IVBag 2044
IVBag - Hierarchy List 2044
IVBag - Member Functions and Data by Group 2044
IVBag - Type Definitions 2044
IVBag - Associated Globals 2045
IVBag - Inherited Member Functions and Data 2045
IVBagAsAvlTree 2047
IVBagAsAvlTree - Hierarchy List 2047
IVBagAsAvlTree - Member Functions and Data by Group 2047
IVBagAsAvlTree - Type Definitions 2048
IVBagAsAvlTree - Associated Globals 2048
IVBagAsAvlTree - Inherited Member Functions and Data 2048
IVBagAsBstTree 2051
IVBagAsBstTree - Hierarchy List. 2051
IVBagAsBstTree - Member Functions and Data by Group 2051
IVBagAsBstTree - Type Definitions 2051
IVBagAsBstTree - Associated Globals 2052
IVBagAsBstTree - Inherited Member Functions and Data 2052
IVBagAsDilTable 2054
IVBagAsDilTable - Hierarchy List 2054
IVBagAsDilTable - Member Functions and Data by Group 2054
IVBagAsDilTable - Type Definitions 2055
IVBagAsDilTable - Associated Globals 2055
IVBagAsDilTable - Inherited Member Functions and Data 2055
IVBagAsHshTable 2058
IVBagAsHshTable - Hierarchy List 2058
IVBagAsHshTable - Member Functions and Data by Group 2058
IVBagAsHshTable - Type Definitions 2058
IVBagAsHshTable - Associated Globals . . . 2059
IVBagAsHshTable - Inherited Member Functions and Data 2059
IVBagAsList 2061
IVBagAsList - Hierarchy List 2061

IVBagAsList - Member Functions and Data by Group 2062
IVBagAsList - Type Definitions 2062
IVBagAsList - Associated Globals 2062
IVBagAsList - Inherited Member Functions and Data 2063
IVBagAsTable 2065
IVBagAsTable - Hierarchy List 2065
IVBagAsTable - Member Functions and Data by Group 2065
IVBagAsTable - Type Definitions 2066
IVBagAsTable - Associated Globals 2066
IVBagAsTable - Inherited Member Functions and Data 2066
IVDeque 2068
IVDeque - Hierarchy List 2069
IVDeque - Member Functions and Data by Group 2069
IVDeque - Type Definitions 2069
IVDeque - Associated Globals 2069
IVDeque - Inherited Member Functions and Data 2070
IVDequeAsDilTable 2072
IVDequeAsDilTable - Hierarchy List 2072
IVDequeAsDilTable - Member Functions and Data by Group 2072
IVDequeAsDilTable - Type Definitions . . . 2072
IVDequeAsDilTable - Associated Globals . . . 2073
IVDequeAsDilTable - Inherited Member Functions and Data 2073
IVDequeAsList 2075
IVDequeAsList - Hierarchy List 2075
IVDequeAsList - Member Functions and Data by Group 2075
IVDequeAsList - Type Definitions 2075
IVDequeAsList - Associated Globals 2076
IVDequeAsList - Inherited Member Functions and Data 2076
IVDequeAsTable 2078
IVDequeAsTable - Hierarchy List 2078
IVDequeAsTable - Member Functions and Data by Group 2078
IVDequeAsTable - Type Definitions. 2079
IVDequeAsTable - Associated Globals 2079
IVDequeAsTable - Inherited Member Functions and Data 2079
IVEqualitySequence 2081
IVEqualitySequence - Hierarchy List 2081
IVEqualitySequence - Member Functions and Data by Group 2082
IVEqualitySequence - Type Definitions . . . 2082
IVEqualitySequence - Associated Globals . . . 2082
IVEqualitySequence - Inherited Member Functions and Data 2083
IVEqualitySequenceAsDilTable 2085
IVEqualitySequenceAsDilTable - Hierarchy List 2085
IVEqualitySequenceAsDilTable - Member Functions and Data by Group 2085
IVEqualitySequenceAsDilTable - Type Definitions 2086
IVEqualitySequenceAsDilTable - Associated Globals 2086
IVEqualitySequenceAsDilTable - Inherited Member Functions and Data 2086
IVEqualitySequenceAsList. 2088
IVEqualitySequenceAsList - Hierarchy List 2089
IVEqualitySequenceAsList - Member Functions and Data by Group 2089
IVEqualitySequenceAsList - Type Definitions 2089
IVEqualitySequenceAsList - Associated Globals 2089
IVEqualitySequenceAsList - Inherited Member Functions and Data 2090
IVEqualitySequenceAsTable 2092
IVEqualitySequenceAsTable - Hierarchy List 2092
IVEqualitySequenceAsTable - Member Functions and Data by Group 2092
IVEqualitySequenceAsTable - Type Definitions 2093
IVEqualitySequenceAsTable - Associated Globals 2093
IVEqualitySequenceAsTable - Inherited Member Functions and Data 2093
IVGBag 2096
IVGBag - Hierarchy List 2096
IVGBag - Member Functions and Data by Group 2096
IVGBag - Type Definitions. 2096
IVGBag - Associated Globals 2097
IVGBag - Inherited Member Functions and Data 2097
IVGBagAsAvlTree 2099
IVGBagAsAvlTree - Hierarchy List 2099
IVGBagAsAvlTree - Member Functions and Data by Group 2099
IVGBagAsAvlTree - Type Definitions 2100
IVGBagAsAvlTree - Associated Globals . . . 2100
IVGBagAsAvlTree - Inherited Member Functions and Data 2100
IVGBagAsBstTree. 2102
IVGBagAsBstTree - Hierarchy List 2103
IVGBagAsBstTree - Member Functions and Data by Group 2103
IVGBagAsBstTree - Type Definitions 2103
IVGBagAsBstTree - Associated Globals . . . 2104
IVGBagAsBstTree - Inherited Member Functions and Data 2104
IVGBagAsDilTable 2106
IVGBagAsDilTable - Hierarchy List. 2106
IVGBagAsDilTable - Member Functions and Data by Group 2106
IVGBagAsDilTable - Type Definitions 2107
IVGBagAsDilTable - Associated Globals . . . 2107
IVGBagAsDilTable - Inherited Member Functions and Data 2107
IVGBagAsHshTable 2109
IVGBagAsHshTable - Hierarchy List 2110
IVGBagAsHshTable - Member Functions and Data by Group 2110
IVGBagAsHshTable - Type Definitions. . . . 2110
IVGBagAsHshTable - Associated Globals . . . 2111
IVGBagAsHshTable - Inherited Member Functions and Data 2111

IVGBagAsList 2113
IVGBagAsList - Hierarchy List 2113
IVGBagAsList - Member Functions and Data by Group 2113
IVGBagAsList - Type Definitions 2114
IVGBagAsList - Associated Globals 2114
IVGBagAsList - Inherited Member Functions and Data 2114
IVGBagAsTable 2116
IVGBagAsTable - Hierarchy List 2116
IVGBagAsTable - Member Functions and Data by Group 2117
IVGBagAsTable - Type Definitions 2117
IVGBagAsTable - Associated Globals 2117
IVGBagAsTable - Inherited Member Functions and Data 2118
IVGDeque 2120
IVGDeque - Hierarchy List 2120
IVGDeque - Member Functions and Data by Group 2120
IVGDeque - Type Definitions 2121
IVGDeque - Associated Globals 2121
IVGDeque - Inherited Member Functions and Data 2121
IVGDequeAsDilTable 2123
IVGDequeAsDilTable - Hierarchy List 2123
IVGDequeAsDilTable - Member Functions and Data by Group 2123
IVGDequeAsDilTable - Type Definitions . . . 2124
IVGDequeAsDilTable - Associated Globals 2124
IVGDequeAsDilTable - Inherited Member Functions and Data 2124
IVGDequeAsList 2126
IVGDequeAsList - Hierarchy List 2126
IVGDequeAsList - Member Functions and Data by Group 2127
IVGDequeAsList - Type Definitions 2127
IVGDequeAsList - Associated Globals 2127
IVGDequeAsList - Inherited Member Functions and Data 2128
IVGDequeAsTable 2129
IVGDequeAsTable - Hierarchy List 2130
IVGDequeAsTable - Member Functions and Data by Group 2130
IVGDequeAsTable - Type Definitions 2130
IVGDequeAsTable - Associated Globals . . . 2131
IVGDequeAsTable - Inherited Member Functions and Data 2131
IVGEqualitySequence 2133
IVGEqualitySequence - Hierarchy List 2133
IVGEqualitySequence - Member Functions and Data by Group 2133
IVGEqualitySequence - Type Definitions . . . 2133
IVGEqualitySequence - Associated Globals 2134
IVGEqualitySequence - Inherited Member Functions and Data 2134
IVGEqualitySequenceAsDilTable 2136
IVGEqualitySequenceAsDilTable - Hierarchy List 2136
IVGEqualitySequenceAsDilTable - Member Functions and Data by Group 2136
IVGEqualitySequenceAsDilTable - Type Definitions 2137
IVGEqualitySequenceAsDilTable - Associated Globals 2137
IVGEqualitySequenceAsDilTable - Inherited Member Functions and Data 2138
IVGEqualitySequenceAsList 2140
IVGEqualitySequenceAsList - Hierarchy List 2140
IVGEqualitySequenceAsList - Member Functions and Data by Group 2140
IVGEqualitySequenceAsList - Type Definitions 2140
IVGEqualitySequenceAsList - Associated Globals 2141
IVGEqualitySequenceAsList - Inherited Member Functions and Data 2141
IVGEqualitySequenceAsTable 2143
IVGEqualitySequenceAsTable - Hierarchy List 2143
IVGEqualitySequenceAsTable - Member Functions and Data by Group 2143
IVGEqualitySequenceAsTable - Type Definitions 2144
IVGEqualitySequenceAsTable - Associated Globals 2144
IVGEqualitySequenceAsTable - Inherited Member Functions and Data 2145
IVGHeap 2147
IVGHeap - Hierarchy List 2147
IVGHeap - Member Functions and Data by Group 2147
IVGHeap - Type Definitions 2147
IVGHeap - Associated Globals 2148
IVGHeap - Inherited Member Functions and Data 2148
IVGHeapAsDilTable 2148
IVGHeapAsDilTable - Hierarchy List 2148
IVGHeapAsDilTable - Member Functions and Data by Group 2149
IVGHeapAsDilTable - Type Definitions . . . 2149
IVGHeapAsDilTable - Associated Globals . . 2149
IVGHeapAsDilTable - Inherited Member Functions and Data 2150
IVGHeapAsList 2150
IVGHeapAsList - Hierarchy List 2150
IVGHeapAsList - Member Functions and Data by Group 2150
IVGHeapAsList - Type Definitions 2151
IVGHeapAsList - Associated Globals 2151
IVGHeapAsList - Inherited Member Functions and Data 2152
IVGHeapAsTable 2152
IVGHeapAsTable - Hierarchy List 2152
IVGHeapAsTable - Member Functions and Data by Group 2152
IVGHeapAsTable - Type Definitions 2153
IVGHeapAsTable - Associated Globals . . . 2153
IVGHeapAsTable - Inherited Member Functions and Data 2153
IVGKeyBag 2154
IVGKeyBag - Hierarchy List 2154
IVGKeyBag - Member Functions and Data by Group 2154

IVGKeyBag - Type Definitions 2155
IVGKeyBag - Associated Globals 2155
IVGKeyBag - Inherited Member Functions and Data 2155
IVGKeyBagAsDilTable 2157
IVGKeyBagAsDilTable - Hierarchy List . . . 2157
IVGKeyBagAsDilTable - Member Functions and Data by Group 2158
IVGKeyBagAsDilTable - Type Definitions. . . 2158
IVGKeyBagAsDilTable - Associated Globals 2158
IVGKeyBagAsDilTable - Inherited Member Functions and Data 2159
IVGKeyBagAsHshTable 2161
IVGKeyBagAsHshTable - Hierarchy List . . . 2161
IVGKeyBagAsHshTable - Member Functions and Data by Group 2161
IVGKeyBagAsHshTable - Type Definitions 2161
IVGKeyBagAsHshTable - Associated Globals 2162
IVGKeyBagAsHshTable - Inherited Member Functions and Data 2162
IVGKeyBagAsList 2164
IVGKeyBagAsList - Hierarchy List 2164
IVGKeyBagAsList - Member Functions and Data by Group 2164
IVGKeyBagAsList - Type Definitions 2165
IVGKeyBagAsList - Associated Globals . . . 2165
IVGKeyBagAsList - Inherited Member Functions and Data 2166
IVGKeyBagAsTable 2168
IVGKeyBagAsTable - Hierarchy List 2168
IVGKeyBagAsTable - Member Functions and Data by Group 2168
IVGKeyBagAsTable - Type Definitions. . . . 2168
IVGKeyBagAsTable - Associated Globals . . . 2169
IVGKeyBagAsTable - Inherited Member Functions and Data 2169
IVGKeySet 2171
IVGKeySet - Hierarchy List 2171
IVGKeySet - Member Functions and Data by Group 2171
IVGKeySet - Type Definitions 2172
IVGKeySet - Associated Globals 2172
IVGKeySet - Inherited Member Functions and Data 2172
IVGKeySetAsAvlTree 2175
IVGKeySetAsAvlTree - Hierarchy List 2175
IVGKeySetAsAvlTree - Member Functions and Data by Group 2175
IVGKeySetAsAvlTree - Type Definitions . . . 2175
IVGKeySetAsAvlTree - Associated Globals . . 2176
IVGKeySetAsAvlTree - Inherited Member Functions and Data 2176
IVGKeySetAsBstTree 2178
IVGKeySetAsBstTree - Hierarchy List 2178
IVGKeySetAsBstTree - Member Functions and Data by Group 2178
IVGKeySetAsBstTree - Type Definitions . . . 2179
IVGKeySetAsBstTree - Associated Globals . . 2179
IVGKeySetAsBstTree - Inherited Member Functions and Data 2180
IVGKeySetAsDilTable 2182
IVGKeySetAsDilTable - Hierarchy List. . . . 2182
IVGKeySetAsDilTable - Member Functions and Data by Group 2182
IVGKeySetAsDilTable - Type Definitions . . . 2182
IVGKeySetAsDilTable - Associated Globals 2183
IVGKeySetAsDilTable - Inherited Member Functions and Data 2183
IVGKeySetAsHshTable 2185
IVGKeySetAsHshTable - Hierarchy List . . . 2185
IVGKeySetAsHshTable - Member Functions and Data by Group 2185
IVGKeySetAsHshTable - Type Definitions . . 2186
IVGKeySetAsHshTable - Associated Globals 2186
IVGKeySetAsHshTable - Inherited Member Functions and Data 2187
IVGKeySetAsList. 2189
IVGKeySetAsList - Hierarchy List 2189
IVGKeySetAsList - Member Functions and Data by Group 2189
IVGKeySetAsList - Type Definitions 2189
IVGKeySetAsList - Associated Globals. . . . 2190
IVGKeySetAsList - Inherited Member Functions and Data 2190
IVGKeySetAsTable 2192
IVGKeySetAsTable - Hierarchy List. 2192
IVGKeySetAsTable - Member Functions and Data by Group 2192
IVGKeySetAsTable - Type Definitions 2193
IVGKeySetAsTable - Associated Globals . . . 2193
IVGKeySetAsTable - Inherited Member Functions and Data 2194
IVGKeySortedBag 2196
IVGKeySortedBag - Hierarchy List 2196
IVGKeySortedBag - Member Functions and Data by Group 2196
IVGKeySortedBag - Type Definitions 2196
IVGKeySortedBag - Associated Globals . . . 2197
IVGKeySortedBag - Inherited Member Functions and Data 2197
IVGKeySortedBagAsDilTable. 2199
IVGKeySortedBagAsDilTable - Hierarchy List 2199
IVGKeySortedBagAsDilTable - Member Functions and Data by Group 2200
IVGKeySortedBagAsDilTable - Type Definitions 2200
IVGKeySortedBagAsDilTable - Associated Globals 2201
IVGKeySortedBagAsDilTable - Inherited Member Functions and Data 2201
IVGKeySortedBagAsList 2203
IVGKeySortedBagAsList - Hierarchy List. . . 2203
IVGKeySortedBagAsList - Member Functions and Data by Group 2204
IVGKeySortedBagAsList - Type Definitions 2204
IVGKeySortedBagAsList - Associated Globals 2204
IVGKeySortedBagAsList - Inherited Member Functions and Data 2205
IVGKeySortedBagAsTable. 2207
IVGKeySortedBagAsTable - Hierarchy List 2207
IVGKeySortedBagAsTable - Member Functions and Data by Group 2207
IVGKeySortedBagAsTable - Type Definitions 2208

IVGKeySortedBagAsTable - Associated Globals 2208
IVGKeySortedBagAsTable - Inherited Member Functions and Data 2209
IVGKeySortedSet 2211
IVGKeySortedSet - Hierarchy List 2211
IVGKeySortedSet - Member Functions and Data by Group 2211
IVGKeySortedSet - Type Definitions 2212
IVGKeySortedSet - Associated Globals . . . 2212
IVGKeySortedSet - Inherited Member Functions and Data 2212
IVGKeySortedSetAsAvlTree 2215
IVGKeySortedSetAsAvlTree - Hierarchy List 2215
IVGKeySortedSetAsAvlTree - Member Functions and Data by Group 2215
IVGKeySortedSetAsAvlTree - Type Definitions 2215
IVGKeySortedSetAsAvlTree - Associated Globals 2216
IVGKeySortedSetAsAvlTree - Inherited Member Functions and Data 2216
IVGKeySortedSetAsBstTree 2218
IVGKeySortedSetAsBstTree - Hierarchy List 2219
IVGKeySortedSetAsBstTree - Member Functions and Data by Group 2219
IVGKeySortedSetAsBstTree - Type Definitions 2219
IVGKeySortedSetAsBstTree - Associated Globals 2220
IVGKeySortedSetAsBstTree - Inherited Member Functions and Data 2220
IVGKeySortedSetAsDilTable 2222
IVGKeySortedSetAsDilTable - Hierarchy List 2222
IVGKeySortedSetAsDilTable - Member Functions and Data by Group 2223
IVGKeySortedSetAsDilTable - Type Definitions 2223
IVGKeySortedSetAsDilTable - Associated Globals 2224
IVGKeySortedSetAsDilTable - Inherited Member Functions and Data 2224
IVGKeySortedSetAsList 2226
IVGKeySortedSetAsList - Hierarchy List . . . 2226
IVGKeySortedSetAsList - Member Functions and Data by Group 2226
IVGKeySortedSetAsList - Type Definitions 2227
IVGKeySortedSetAsList - Associated Globals 2227
IVGKeySortedSetAsList - Inherited Member Functions and Data 2228
IVGKeySortedSetAsTable 2230
IVGKeySortedSetAsTable - Hierarchy List . . 2230
IVGKeySortedSetAsTable - Member Functions and Data by Group 2230
IVGKeySortedSetAsTable - Type Definitions 2231
IVGKeySortedSetAsTable - Associated Globals 2231
IVGKeySortedSetAsTable - Inherited Member Functions and Data 2232
IVGMap. 2234
IVGMap - Hierarchy List 2234
IVGMap - Member Functions and Data by Group 2234
IVGMap - Type Definitions 2235
IVGMap - Associated Globals 2235
IVGMap - Inherited Member Functions and Data 2235
IVGMapAsAvlTree 2238
IVGMapAsAvlTree - Hierarchy List. 2238
IVGMapAsAvlTree - Member Functions and Data by Group 2238
IVGMapAsAvlTree - Type Definitions 2239
IVGMapAsAvlTree - Associated Globals . . . 2239
IVGMapAsAvlTree - Inherited Member Functions and Data 2239
IVGMapAsBstTree 2242
IVGMapAsBstTree - Hierarchy List. 2242
IVGMapAsBstTree - Member Functions and Data by Group 2242
IVGMapAsBstTree - Type Definitions 2242
IVGMapAsBstTree - Associated Globals . . . 2243
IVGMapAsBstTree - Inherited Member Functions and Data 2243
IVGMapAsDilTable 2245
IVGMapAsDilTable - Hierarchy List 2245
IVGMapAsDilTable - Member Functions and Data by Group 2246
IVGMapAsDilTable - Type Definitions. . . . 2246
IVGMapAsDilTable - Associated Globals . . . 2247
IVGMapAsDilTable - Inherited Member Functions and Data 2247
IVGMapAsHshTable 2249
IVGMapAsHshTable - Hierarchy List 2249
IVGMapAsHshTable - Member Functions and Data by Group 2249
IVGMapAsHshTable - Type Definitions . . . 2250
IVGMapAsHshTable - Associated Globals . . 2250
IVGMapAsHshTable - Inherited Member Functions and Data 2251
IVGMapAsList 2253
IVGMapAsList - Hierarchy List 2253
IVGMapAsList - Member Functions and Data by Group 2253
IVGMapAsList - Type Definitions 2254
IVGMapAsList - Associated Globals 2254
IVGMapAsList - Inherited Member Functions and Data 2254
IVGMapAsTable 2257
IVGMapAsTable - Hierarchy List 2257
IVGMapAsTable - Member Functions and Data by Group 2257
IVGMapAsTable - Type Definitions. 2258
IVGMapAsTable - Associated Globals 2258
IVGMapAsTable - Inherited Member Functions and Data 2258
IVGMultiwayTree 2261
IVGMultiwayTree - Hierarchy List 2261
IVGMultiwayTree - Member Functions and Data by Group 2261
IVGMultiwayTree - Type Definitions 2261
IVGMultiwayTree - Associated Globals . . . 2262
IVGMultiwayTree - Inherited Member Functions and Data 2262
IVGPriorityQueue 2264
IVGPriorityQueue - Hierarchy List 2264

IVGPriorityQueue - Member Functions and Data by Group 2264
IVGPriorityQueue - Type Definitions 2265
IVGPriorityQueue - Associated Globals . . . 2265
IVGPriorityQueue - Inherited Member Functions and Data 2265
IVGPriorityQueueAsDilTable 2267
IVGPriorityQueueAsDilTable - Hierarchy List 2267
IVGPriorityQueueAsDilTable - Member Functions and Data by Group 2268
IVGPriorityQueueAsDilTable - Type Definitions 2268
IVGPriorityQueueAsDilTable - Associated Globals 2268
IVGPriorityQueueAsDilTable - Inherited Member Functions and Data 2269
IVGPriorityQueueAsList 2271
IVGPriorityQueueAsList - Hierarchy List. . . 2271
IVGPriorityQueueAsList - Member Functions and Data by Group 2271
IVGPriorityQueueAsList - Type Definitions 2271
IVGPriorityQueueAsList - Associated Globals 2272
IVGPriorityQueueAsList - Inherited Member Functions and Data 2272
IVGPriorityQueueAsTable 2274
IVGPriorityQueueAsTable - Hierarchy List 2274
IVGPriorityQueueAsTable - Member Functions and Data by Group 2274
IVGPriorityQueueAsTable - Type Definitions 2275
IVGPriorityQueueAsTable - Associated Globals 2275
IVGPriorityQueueAsTable - Inherited Member Functions and Data 2275
IVGQueue 2277
IVGQueue - Hierarchy List 2277
IVGQueue - Member Functions and Data by Group 2278
IVGQueue - Type Definitions. 2278
IVGQueue - Associated Globals 2278
IVGQueue - Inherited Member Functions and Data 2279
IVGQueueAsDilTable 2280
IVGQueueAsDilTable - Hierarchy List. . . . 2281
IVGQueueAsDilTable - Member Functions and Data by Group 2281
IVGQueueAsDilTable - Type Definitions . . . 2281
IVGQueueAsDilTable - Associated Globals 2282
IVGQueueAsDilTable - Inherited Member Functions and Data 2282
IVGQueueAsList 2284
IVGQueueAsList - Hierarchy List 2284
IVGQueueAsList - Member Functions and Data by Group 2284
IVGQueueAsList - Type Definitions 2285
IVGQueueAsList - Associated Globals. . . . 2285
IVGQueueAsList - Inherited Member Functions and Data 2285
IVGQueueAsTable 2287
IVGQueueAsTable - Hierarchy List. 2287
IVGQueueAsTable - Member Functions and Data by Group 2287
IVGQueueAsTable - Type Definitions 2288
IVGQueueAsTable - Associated Globals . . . 2288
IVGQueueAsTable - Inherited Member Functions and Data 2288
IVGRelation 2290
IVGRelation - Hierarchy List 2290
IVGRelation - Member Functions and Data by Group 2291
IVGRelation - Type Definitions 2291
IVGRelation - Associated Globals 2291
IVGRelation - Inherited Member Functions and Data 2292
IVGRelationAsDilTable 2294
IVGRelationAsDilTable - Hierarchy List . . . 2294
IVGRelationAsDilTable - Member Functions and Data by Group 2294
IVGRelationAsDilTable - Type Definitions . . 2295
IVGRelationAsDilTable - Associated Globals 2295
IVGRelationAsDilTable - Inherited Member Functions and Data 2295
IVGRelationAsHshTable 2298
IVGRelationAsHshTable - Hierarchy List . . . 2298
IVGRelationAsHshTable - Member Functions and Data by Group 2298
IVGRelationAsHshTable - Type Definitions 2298
IVGRelationAsHshTable - Associated Globals 2299
IVGRelationAsHshTable - Inherited Member Functions and Data 2299
IVGRelationAsList 2301
IVGRelationAsList - Hierarchy List. 2302
IVGRelationAsList - Member Functions and Data by Group 2302
IVGRelationAsList - Type Definitions 2302
IVGRelationAsList - Associated Globals . . . 2303
IVGRelationAsList - Inherited Member Functions and Data 2303
IVGRelationAsTable 2305
IVGRelationAsTable - Hierarchy List 2305
IVGRelationAsTable - Member Functions and Data by Group 2306
IVGRelationAsTable - Type Definitions . . . 2306
IVGRelationAsTable - Associated Globals. . . 2306
IVGRelationAsTable - Inherited Member Functions and Data 2307
IVGSequence 2309
IVGSequence - Hierarchy List 2309
IVGSequence - Member Functions and Data by Group 2309
IVGSequence - Type Definitions. 2310
IVGSequence - Associated Globals 2310
IVGSequence - Inherited Member Functions and Data 2311
IVGSequenceAsDilTable 2313
IVGSequenceAsDilTable - Hierarchy List . . . 2313
IVGSequenceAsDilTable - Member Functions and Data by Group 2313
IVGSequenceAsDilTable - Type Definitions 2314
IVGSequenceAsDilTable - Associated Globals 2314
IVGSequenceAsDilTable - Inherited Member Functions and Data 2315
IVGSequenceAsList 2317
IVGSequenceAsList - Hierarchy List 2317

IVGSequenceAsList - Member Functions and Data by Group 2317
IVGSequenceAsList - Type Definitions. . . . 2318
IVGSequenceAsList - Associated Globals . . . 2318
IVGSequenceAsList - Inherited Member Functions and Data 2319
IVGSequenceAsTable 2321
IVGSequenceAsTable - Hierarchy List 2321
IVGSequenceAsTable - Member Functions and Data by Group 2321
IVGSequenceAsTable - Type Definitions . . . 2322
IVGSequenceAsTable - Associated Globals . . 2322
IVGSequenceAsTable - Inherited Member Functions and Data 2323
IVGSet 2325
IVGSet - Hierarchy List 2325
IVGSet - Member Functions and Data by Group 2325
IVGSet - Type Definitions 2326
IVGSet - Associated Globals 2326
IVGSet - Inherited Member Functions and Data 2327
IVGSetAsAvlTree. 2329
IVGSetAsAvlTree - Hierarchy List 2329
IVGSetAsAvlTree - Member Functions and Data by Group 2329
IVGSetAsAvlTree - Type Definitions 2329
IVGSetAsAvlTree - Associated Globals. . . . 2330
IVGSetAsAvlTree - Inherited Member Functions and Data 2330
IVGSetAsBstTree 2332
IVGSetAsBstTree - Hierarchy List 2332
IVGSetAsBstTree - Member Functions and Data by Group 2332
IVGSetAsBstTree - Type Definitions 2333
IVGSetAsBstTree - Associated Globals. . . . 2333
IVGSetAsBstTree - Inherited Member Functions and Data 2334
IVGSetAsDilTable 2336
IVGSetAsDilTable - Hierarchy List 2336
IVGSetAsDilTable - Member Functions and Data by Group 2336
IVGSetAsDilTable - Type Definitions 2336
IVGSetAsDilTable - Associated Globals . . . 2337
IVGSetAsDilTable - Inherited Member Functions and Data 2337
IVGSetAsHshTable 2339
IVGSetAsHshTable - Hierarchy List 2339
IVGSetAsHshTable - Member Functions and Data by Group 2339
IVGSetAsHshTable - Type Definitions 2340
IVGSetAsHshTable - Associated Globals . . . 2340
IVGSetAsHshTable - Inherited Member Functions and Data 2341
IVGSetAsList 2343
IVGSetAsList - Hierarchy List 2343
IVGSetAsList - Member Functions and Data by Group 2343
IVGSetAsList - Type Definitions. 2343
IVGSetAsList - Associated Globals 2344
IVGSetAsList - Inherited Member Functions and Data 2344
IVGSetAsTable 2346
IVGSetAsTable - Hierarchy List 2346
IVGSetAsTable - Member Functions and Data by Group 2346
IVGSetAsTable - Type Definitions 2347
IVGSetAsTable - Associated Globals 2347
IVGSetAsTable - Inherited Member Functions and Data 2348
IVGSortedBag. 2350
IVGSortedBag - Hierarchy List 2350
IVGSortedBag - Member Functions and Data by Group 2350
IVGSortedBag - Type Definitions 2350
IVGSortedBag - Associated Globals. 2351
IVGSortedBag - Inherited Member Functions and Data 2351
IVGSortedBagAsAvlTree 2353
IVGSortedBagAsAvlTree - Hierarchy List. . . 2354
IVGSortedBagAsAvlTree - Member Functions and Data by Group 2354
IVGSortedBagAsAvlTree - Type Definitions 2354
IVGSortedBagAsAvlTree - Associated Globals 2355
IVGSortedBagAsAvlTree - Inherited Member Functions and Data 2355
IVGSortedBagAsBstTree 2357
IVGSortedBagAsBstTree - Hierarchy List . . . 2357
IVGSortedBagAsBstTree - Member Functions and Data by Group 2358
IVGSortedBagAsBstTree - Type Definitions 2358
IVGSortedBagAsBstTree - Associated Globals 2359
IVGSortedBagAsBstTree - Inherited Member Functions and Data 2359
IVGSortedBagAsDilTable 2361
IVGSortedBagAsDilTable - Hierarchy List . . 2361
IVGSortedBagAsDilTable - Member Functions and Data by Group 2362
IVGSortedBagAsDilTable - Type Definitions 2362
IVGSortedBagAsDilTable - Associated Globals 2362
IVGSortedBagAsDilTable - Inherited Member Functions and Data 2363
IVGSortedBagAsList. 2365
IVGSortedBagAsList - Hierarchy List 2365
IVGSortedBagAsList - Member Functions and Data by Group 2365
IVGSortedBagAsList - Type Definitions . . . 2366
IVGSortedBagAsList - Associated Globals . . 2366
IVGSortedBagAsList - Inherited Member Functions and Data 2367
IVGSortedBagAsTable 2369
IVGSortedBagAsTable - Hierarchy List . . . 2369
IVGSortedBagAsTable - Member Functions and Data by Group 2369
IVGSortedBagAsTable - Type Definitions . . . 2370
IVGSortedBagAsTable - Associated Globals 2370
IVGSortedBagAsTable - Inherited Member Functions and Data 2370
IVGSortedMap 2373
IVGSortedMap - Hierarchy List 2373
IVGSortedMap - Member Functions and Data by Group 2373
IVGSortedMap - Type Definitions 2373

IVGSortedMap - Associated Globals 2374
IVGSortedMap - Inherited Member Functions and Data 2374
IVGSortedMapAsAvlTree 2377
IVGSortedMapAsAvlTree - Hierarchy List . . 2377
IVGSortedMapAsAvlTree - Member Functions and Data by Group 2377
IVGSortedMapAsAvlTree - Type Definitions 2377
IVGSortedMapAsAvlTree - Associated Globals 2378
IVGSortedMapAsAvlTree - Inherited Member Functions and Data 2378
IVGSortedMapAsBstTree 2380
IVGSortedMapAsBstTree - Hierarchy List . . 2381
IVGSortedMapAsBstTree - Member Functions and Data by Group 2381
IVGSortedMapAsBstTree - Type Definitions 2381
IVGSortedMapAsBstTree - Associated Globals 2382
IVGSortedMapAsBstTree - Inherited Member Functions and Data 2382
IVGSortedMapAsDilTable 2384
IVGSortedMapAsDilTable - Hierarchy List 2384
IVGSortedMapAsDilTable - Member Functions and Data by Group 2385
IVGSortedMapAsDilTable - Type Definitions 2385
IVGSortedMapAsDilTable - Associated Globals 2386
IVGSortedMapAsDilTable - Inherited Member Functions and Data 2386
IVGSortedMapAsList 2388
IVGSortedMapAsList - Hierarchy List 2388
IVGSortedMapAsList - Member Functions and Data by Group 2389
IVGSortedMapAsList - Type Definitions . . . 2389
IVGSortedMapAsList - Associated Globals 2389
IVGSortedMapAsList - Inherited Member Functions and Data 2390
IVGSortedMapAsTable 2392
IVGSortedMapAsTable - Hierarchy List . . . 2392
IVGSortedMapAsTable - Member Functions and Data by Group 2392
IVGSortedMapAsTable - Type Definitions . . 2393
IVGSortedMapAsTable - Associated Globals 2393
IVGSortedMapAsTable - Inherited Member Functions and Data 2394
IVGSortedRelation 2396
IVGSortedRelation - Hierarchy List. 2396
IVGSortedRelation - Member Functions and Data by Group 2396
IVGSortedRelation - Type Definitions 2397
IVGSortedRelation - Associated Globals . . . 2397
IVGSortedRelation - Inherited Member Functions and Data 2397
IVGSortedRelationAsDilTable 2400
IVGSortedRelationAsDilTable - Hierarchy List 2400
IVGSortedRelationAsDilTable - Member Functions and Data by Group 2400
IVGSortedRelationAsDilTable - Type Definitions 2401
IVGSortedRelationAsDilTable - Associated Globals 2401
IVGSortedRelationAsDilTable - Inherited Member Functions and Data 2401
IVGSortedRelationAsTable 2404
IVGSortedRelationAsTable - Hierarchy List 2404
IVGSortedRelationAsTable - Member Functions and Data by Group 2404
IVGSortedRelationAsTable - Type Definitions 2404
IVGSortedRelationAsTable - Associated Globals 2405
IVGSortedRelationAsTable - Inherited Member Functions and Data 2405
IVGSortedSet 2407
IVGSortedSet - Hierarchy List 2408
IVGSortedSet - Member Functions and Data by Group 2408
IVGSortedSet - Type Definitions. 2408
IVGSortedSet - Associated Globals 2409
IVGSortedSet - Inherited Member Functions and Data 2409
IVGSortedSetAsAvlTree. 2411
IVGSortedSetAsAvlTree - Hierarchy List . . . 2411
IVGSortedSetAsAvlTree - Member Functions and Data by Group 2412
IVGSortedSetAsAvlTree - Type Definitions 2412
IVGSortedSetAsAvlTree - Associated Globals 2413
IVGSortedSetAsAvlTree - Inherited Member Functions and Data 2413
IVGSortedSetAsBstTree. 2415
IVGSortedSetAsBstTree - Hierarchy List . . . 2415
IVGSortedSetAsBstTree - Member Functions and Data by Group 2416
IVGSortedSetAsBstTree - Type Definitions . . 2416
IVGSortedSetAsBstTree - Associated Globals 2416
IVGSortedSetAsBstTree - Inherited Member Functions and Data 2417
IVGSortedSetAsDilTable 2419
IVGSortedSetAsDilTable - Hierarchy List. . . 2419
IVGSortedSetAsDilTable - Member Functions and Data by Group 2419
IVGSortedSetAsDilTable - Type Definitions 2420
IVGSortedSetAsDilTable - Associated Globals 2420
IVGSortedSetAsDilTable - Inherited Member Functions and Data 2421
IVGSortedSetAsList 2423
IVGSortedSetAsList - Hierarchy List 2423
IVGSortedSetAsList - Member Functions and Data by Group 2423
IVGSortedSetAsList - Type Definitions . . . 2424
IVGSortedSetAsList - Associated Globals. . . 2424
IVGSortedSetAsList - Inherited Member Functions and Data 2424
IVGSortedSetAsTable 2427
IVGSortedSetAsTable - Hierarchy List. . . . 2427
IVGSortedSetAsTable - Member Functions and Data by Group 2427
IVGSortedSetAsTable - Type Definitions . . . 2428
IVGSortedSetAsTable - Associated Globals 2428
IVGSortedSetAsTable - Inherited Member Functions and Data 2428
IVGStack 2431
IVGStack - Hierarchy List 2431
IVGStack - Member Functions and Data by Group 2431
IVGStack - Type Definitions 2431

IVGStack - Associated Globals 2432
IVGStack - Inherited Member Functions and Data 2432
IVGStackAsDilTable 2434
IVGStackAsDilTable - Hierarchy List 2434
IVGStackAsDilTable - Member Functions and Data by Group 2434
IVGStackAsDilTable - Type Definitions . . . 2435
IVGStackAsDilTable - Associated Globals. . . 2435
IVGStackAsDilTable - Inherited Member Functions and Data 2435
IVGStackAsList 2437
IVGStackAsList - Hierarchy List. 2437
IVGStackAsList - Member Functions and Data by Group 2437
IVGStackAsList - Type Definitions 2438
IVGStackAsList - Associated Globals 2438
IVGStackAsList - Inherited Member Functions and Data 2439
IVGStackAsTable 2440
IVGStackAsTable - Hierarchy List 2440
IVGStackAsTable - Member Functions and Data by Group 2441
IVGStackAsTable - Type Definitions 2441
IVGStackAsTable - Associated Globals. . . . 2442
IVGStackAsTable - Inherited Member Functions and Data 2442
IVHeap 2444
IVHeap - Hierarchy List 2444
IVHeap - Member Functions and Data by Group 2444
IVHeap - Type Definitions. 2444
IVHeap - Associated Globals 2444
IVHeap - Inherited Member Functions and Data 2445
IVHeapAsDilTable 2445
IVHeapAsDilTable - Hierarchy List. 2446
IVHeapAsDilTable - Member Functions and Data by Group 2446
IVHeapAsDilTable - Type Definitions 2446
IVHeapAsDilTable - Associated Globals . . . 2446
IVHeapAsDilTable - Inherited Member Functions and Data 2447
IVHeapAsList 2447
IVHeapAsList - Hierarchy List 2447
IVHeapAsList - Member Functions and Data by Group 2448
IVHeapAsList - Type Definitions 2448
IVHeapAsList - Associated Globals. 2448
IVHeapAsList - Inherited Member Functions and Data 2449
IVHeapAsTable 2449
IVHeapAsTable - Hierarchy List. 2449
IVHeapAsTable - Member Functions and Data by Group 2450
IVHeapAsTable - Type Definitions 2450
IVHeapAsTable - Associated Globals 2450
IVHeapAsTable - Inherited Member Functions and Data 2451
IVKeyBag 2451
IVKeyBag - Hierarchy List 2451
IVKeyBag - Member Functions and Data by Group 2452
IVKeyBag - Type Definitions 2452
IVKeyBag - Associated Globals 2452
IVKeyBag - Inherited Member Functions and Data 2453
IVKeyBagAsDilTable 2455
IVKeyBagAsDilTable - Hierarchy List 2455
IVKeyBagAsDilTable - Member Functions and Data by Group 2455
IVKeyBagAsDilTable - Type Definitions . . . 2456
IVKeyBagAsDilTable - Associated Globals . . 2456
IVKeyBagAsDilTable - Inherited Member Functions and Data 2456
IVKeyBagAsHshTable 2458
IVKeyBagAsHshTable - Hierarchy List . . . 2458
IVKeyBagAsHshTable - Member Functions and Data by Group 2459
IVKeyBagAsHshTable - Type Definitions . . . 2459
IVKeyBagAsHshTable - Associated Globals 2459
IVKeyBagAsHshTable - Inherited Member Functions and Data 2460
IVKeyBagAsList 2462
IVKeyBagAsList - Hierarchy List 2462
IVKeyBagAsList - Member Functions and Data by Group 2462
IVKeyBagAsList - Type Definitions. 2463
IVKeyBagAsList - Associated Globals 2463
IVKeyBagAsList - Inherited Member Functions and Data 2463
IVKeyBagAsTable 2465
IVKeyBagAsTable - Hierarchy List 2465
IVKeyBagAsTable - Member Functions and Data by Group 2466
IVKeyBagAsTable - Type Definitions 2466
IVKeyBagAsTable - Associated Globals . . . 2466
IVKeyBagAsTable - Inherited Member Functions and Data 2467
IVKeySet 2469
IVKeySet - Hierarchy List 2469
IVKeySet - Member Functions and Data by Group 2469
IVKeySet - Type Definitions 2470
IVKeySet - Associated Globals 2470
IVKeySet - Inherited Member Functions and Data 2470
IVKeySetAsAvlTree 2472
IVKeySetAsAvlTree - Hierarchy List 2473
IVKeySetAsAvlTree - Member Functions and Data by Group 2473
IVKeySetAsAvlTree - Type Definitions. . . . 2473
IVKeySetAsAvlTree - Associated Globals . . . 2473
IVKeySetAsAvlTree - Inherited Member Functions and Data 2474
IVKeySetAsBstTree 2476
IVKeySetAsBstTree - Hierarchy List 2476
IVKeySetAsBstTree - Member Functions and Data by Group 2476
IVKeySetAsBstTree - Type Definitions 2477
IVKeySetAsBstTree - Associated Globals . . . 2477

IVKeySetAsBstTree - Inherited Member Functions and Data 2477
IVKeySetAsDilTable 2479
IVKeySetAsDilTable - Hierarchy List 2479
IVKeySetAsDilTable - Member Functions and Data by Group 2480
IVKeySetAsDilTable - Type Definitions . . . 2480
IVKeySetAsDilTable - Associated Globals. . . 2480
IVKeySetAsDilTable - Inherited Member Functions and Data 2481
IVKeySetAsHshTable 2483
IVKeySetAsHshTable - Hierarchy List 2483
IVKeySetAsHshTable - Member Functions and Data by Group 2483
IVKeySetAsHshTable - Type Definitions . . . 2483
IVKeySetAsHshTable - Associated Globals 2484
IVKeySetAsHshTable - Inherited Member Functions and Data 2484
IVKeySetAsList 2486
IVKeySetAsList - Hierarchy List. 2486
IVKeySetAsList - Member Functions and Data by Group 2487
IVKeySetAsList - Type Definitions 2487
IVKeySetAsList - Associated Globals 2487
IVKeySetAsList - Inherited Member Functions and Data 2488
IVKeySetAsTable 2490
IVKeySetAsTable - Hierarchy List 2490
IVKeySetAsTable - Member Functions and Data by Group 2490
IVKeySetAsTable - Type Definitions 2490
IVKeySetAsTable - Associated Globals. . . . 2491
IVKeySetAsTable - Inherited Member Functions and Data 2491
IVKeySortedBag 2493
IVKeySortedBag - Hierarchy List 2493
IVKeySortedBag - Member Functions and Data by Group 2494
IVKeySortedBag - Type Definitions. 2494
IVKeySortedBag - Associated Globals 2494
IVKeySortedBag - Inherited Member Functions and Data 2495
IVKeySortedBagAsDilTable 2497
IVKeySortedBagAsDilTable - Hierarchy List 2497
IVKeySortedBagAsDilTable - Member Functions and Data by Group 2498
IVKeySortedBagAsDilTable - Type Definitions 2498
IVKeySortedBagAsDilTable - Associated Globals 2498
IVKeySortedBagAsDilTable - Inherited Member Functions and Data 2499
IVKeySortedBagAsList 2501
IVKeySortedBagAsList - Hierarchy List . . . 2501
IVKeySortedBagAsList - Member Functions and Data by Group 2502
IVKeySortedBagAsList - Type Definitions . . 2502
IVKeySortedBagAsList - Associated Globals 2502
IVKeySortedBagAsList - Inherited Member Functions and Data 2503
IVKeySortedBagAsTable 2505
IVKeySortedBagAsTable - Hierarchy List . . . 2505
IVKeySortedBagAsTable - Member Functions and Data by Group 2506
IVKeySortedBagAsTable - Type Definitions 2506
IVKeySortedBagAsTable - Associated Globals 2506
IVKeySortedBagAsTable - Inherited Member Functions and Data 2507
IVKeySortedSet 2509
IVKeySortedSet - Hierarchy List. 2509
IVKeySortedSet - Member Functions and Data by Group 2510
IVKeySortedSet - Type Definitions 2510
IVKeySortedSet - Associated Globals 2510
IVKeySortedSet - Inherited Member Functions and Data 2510
IVKeySortedSetAsAvlTree 2513
IVKeySortedSetAsAvlTree - Hierarchy List 2513
IVKeySortedSetAsAvlTree - Member Functions and Data by Group 2513
IVKeySortedSetAsAvlTree - Type Definitions 2514
IVKeySortedSetAsAvlTree - Associated Globals 2514
IVKeySortedSetAsAvlTree - Inherited Member Functions and Data 2514
IVKeySortedSetAsBstTree 2517
IVKeySortedSetAsBstTree - Hierarchy List . . 2517
IVKeySortedSetAsBstTree - Member Functions and Data by Group 2517
IVKeySortedSetAsBstTree - Type Definitions 2518
IVKeySortedSetAsBstTree - Associated Globals 2518
IVKeySortedSetAsBstTree - Inherited Member Functions and Data 2518
IVKeySortedSetAsDilTable 2521
IVKeySortedSetAsDilTable - Hierarchy List 2521
IVKeySortedSetAsDilTable - Member Functions and Data by Group 2521
IVKeySortedSetAsDilTable - Type Definitions 2522
IVKeySortedSetAsDilTable - Associated Globals 2522
IVKeySortedSetAsDilTable - Inherited Member Functions and Data 2522
IVKeySortedSetAsList 2525
IVKeySortedSetAsList - Hierarchy List . . . 2525
IVKeySortedSetAsList - Member Functions and Data by Group 2525
IVKeySortedSetAsList - Type Definitions . . . 2525
IVKeySortedSetAsList - Associated Globals 2526
IVKeySortedSetAsList - Inherited Member Functions and Data 2526
IVKeySortedSetAsTable. 2528
IVKeySortedSetAsTable - Hierarchy List . . . 2528
IVKeySortedSetAsTable - Member Functions and Data by Group 2529
IVKeySortedSetAsTable - Type Definitions . . 2529
IVKeySortedSetAsTable - Associated Globals 2529
IVKeySortedSetAsTable - Inherited Member Functions and Data 2530
IVMap 2532
IVMap - Hierarchy List. 2532
IVMap - Member Functions and Data by Group 2533
IVMap - Type Definitions 2533
IVMap - Associated Globals 2533
IVMap - Inherited Member Functions and Data 2533

IVMapAsAvlTree 2536
IVMapAsAvlTree - Hierarchy List 2536
IVMapAsAvlTree - Member Functions and Data by Group 2536
IVMapAsAvlTree - Type Definitions 2537
IVMapAsAvlTree - Associated Globals. . . . 2537
IVMapAsAvlTree - Inherited Member Functions and Data 2537
IVMapAsBstTree 2540
IVMapAsBstTree - Hierarchy List 2540
IVMapAsBstTree - Member Functions and Data by Group 2540
IVMapAsBstTree - Type Definitions 2541
IVMapAsBstTree - Associated Globals 2541
IVMapAsBstTree - Inherited Member Functions and Data 2541
IVMapAsDilTable 2544
IVMapAsDilTable - Hierarchy List 2544
IVMapAsDilTable - Member Functions and Data by Group 2544
IVMapAsDilTable - Type Definitions 2545
IVMapAsDilTable - Associated Globals . . . 2545
IVMapAsDilTable - Inherited Member Functions and Data 2545
IVMapAsHshTable 2548
IVMapAsHshTable - Hierarchy List 2548
IVMapAsHshTable - Member Functions and Data by Group 2548
IVMapAsHshTable - Type Definitions 2548
IVMapAsHshTable - Associated Globals . . . 2549
IVMapAsHshTable - Inherited Member Functions and Data 2549
IVMapAsList 2552
IVMapAsList - Hierarchy List 2552
IVMapAsList - Member Functions and Data by Group 2552
IVMapAsList - Type Definitions 2552
IVMapAsList - Associated Globals 2553
IVMapAsList - Inherited Member Functions and Data 2553
IVMapAsTable 2556
IVMapAsTable - Hierarchy List 2556
IVMapAsTable - Member Functions and Data by Group 2556
IVMapAsTable - Type Definitions 2556
IVMapAsTable - Associated Globals 2557
IVMapAsTable - Inherited Member Functions and Data 2557
IVMultiwayTree 2560
IVMultiwayTree - Hierarchy List 2560
IVMultiwayTree - Member Functions and Data by Group 2560
IVMultiwayTree - Type Definitions 2560
IVMultiwayTree - Associated Globals 2561
IVMultiwayTree - Inherited Member Functions and Data 2561
IVPriorityQueue 2563
IVPriorityQueue - Hierarchy List 2563
IVPriorityQueue - Member Functions and Data by Group 2563
IVPriorityQueue - Type Definitions. 2564
IVPriorityQueue - Associated Globals 2564
IVPriorityQueue - Inherited Member Functions and Data 2564
IVPriorityQueueAsDilTable 2566
IVPriorityQueueAsDilTable - Hierarchy List 2566
IVPriorityQueueAsDilTable - Member Functions and Data by Group 2567
IVPriorityQueueAsDilTable - Type Definitions 2567
IVPriorityQueueAsDilTable - Associated Globals 2567
IVPriorityQueueAsDilTable - Inherited Member Functions and Data 2568
IVPriorityQueueAsList 2570
IVPriorityQueueAsList - Hierarchy List . . . 2570
IVPriorityQueueAsList - Member Functions and Data by Group 2570
IVPriorityQueueAsList - Type Definitions . . 2570
IVPriorityQueueAsList - Associated Globals 2571
IVPriorityQueueAsList - Inherited Member Functions and Data 2571
IVPriorityQueueAsTable 2573
IVPriorityQueueAsTable - Hierarchy List. . . 2573
IVPriorityQueueAsTable - Member Functions and Data by Group 2573
IVPriorityQueueAsTable - Type Definitions 2574
IVPriorityQueueAsTable - Associated Globals 2574
IVPriorityQueueAsTable - Inherited Member Functions and Data 2574
IVQueue 2577
IVQueue - Hierarchy List 2577
IVQueue - Member Functions and Data by Group 2577
IVQueue - Type Definitions 2577
IVQueue - Associated Globals 2578
IVQueue - Inherited Member Functions and Data 2578
IVQueueAsDilTable 2580
IVQueueAsDilTable - Hierarchy List 2580
IVQueueAsDilTable - Member Functions and Data by Group 2580
IVQueueAsDilTable - Type Definitions . . . 2580
IVQueueAsDilTable - Associated Globals. . . 2581
IVQueueAsDilTable - Inherited Member Functions and Data 2581
IVQueueAsList 2583
IVQueueAsList - Hierarchy List 2583
IVQueueAsList - Member Functions and Data by Group 2583
IVQueueAsList - Type Definitions 2584
IVQueueAsList - Associated Globals 2584
IVQueueAsList - Inherited Member Functions and Data 2584
IVQueueAsTable 2586
IVQueueAsTable - Hierarchy List 2586
IVQueueAsTable - Member Functions and Data by Group 2587
IVQueueAsTable - Type Definitions 2587
IVQueueAsTable - Associated Globals 2587
IVQueueAsTable - Inherited Member Functions and Data 2588
IVRelation 2590

IVRelation - Hierarchy List 2590
IVRelation - Member Functions and Data by Group 2590
IVRelation - Type Definitions. 2590
IVRelation - Associated Globals 2591
IVRelation - Inherited Member Functions and Data 2591
IVRelationAsDilTable 2593
IVRelationAsDilTable - Hierarchy List. . . . 2593
IVRelationAsDilTable - Member Functions and Data by Group 2594
IVRelationAsDilTable - Type Definitions . . . 2594
IVRelationAsDilTable - Associated Globals 2594
IVRelationAsDilTable - Inherited Member Functions and Data 2595
IVRelationAsHshTable 2597
IVRelationAsHshTable - Hierarchy List . . . 2597
IVRelationAsHshTable - Member Functions and Data by Group 2598
IVRelationAsHshTable - Type Definitions. . . 2598
IVRelationAsHshTable - Associated Globals 2598
IVRelationAsHshTable - Inherited Member Functions and Data 2598
IVRelationAsList 2601
IVRelationAsList - Hierarchy List 2601
IVRelationAsList - Member Functions and Data by Group 2601
IVRelationAsList - Type Definitions 2602
IVRelationAsList - Associated Globals. . . . 2602
IVRelationAsList - Inherited Member Functions and Data 2602
IVRelationAsTable 2605
IVRelationAsTable - Hierarchy List. 2605
IVRelationAsTable - Member Functions and Data by Group 2605
IVRelationAsTable - Type Definitions 2605
IVRelationAsTable - Associated Globals . . . 2606
IVRelationAsTable - Inherited Member Functions and Data 2606
IVSequence. 2609
IVSequence - Hierarchy List 2609
IVSequence - Member Functions and Data by Group 2609
IVSequence - Type Definitions 2609
IVSequence - Associated Globals 2610
IVSequence - Inherited Member Functions and Data 2610
IVSequenceAsDilTable 2612
IVSequenceAsDilTable - Hierarchy List . . . 2612
IVSequenceAsDilTable - Member Functions and Data by Group 2613
IVSequenceAsDilTable - Type Definitions. . . 2613
IVSequenceAsDilTable - Associated Globals 2613
IVSequenceAsDilTable - Inherited Member Functions and Data 2614
IVSequenceAsList 2616
IVSequenceAsList - Hierarchy List 2616
IVSequenceAsList - Member Functions and Data by Group 2617
IVSequenceAsList - Type Definitions 2617
IVSequenceAsList - Associated Globals . . . 2617
IVSequenceAsList - Inherited Member Functions and Data 2618
IVSequenceAsTable 2620
IVSequenceAsTable - Hierarchy List 2620
IVSequenceAsTable - Member Functions and Data by Group 2621
IVSequenceAsTable - Type Definitions. . . . 2621
IVSequenceAsTable - Associated Globals . . . 2621
IVSequenceAsTable - Inherited Member Functions and Data 2622
IVSet. 2624
IVSet - Hierarchy List 2624
IVSet - Member Functions and Data by Group 2625
IVSet - Type Definitions 2625
IVSet - Associated Globals. 2625
IVSet - Inherited Member Functions and Data 2626
IVSetAsAvlTree 2628
IVSetAsAvlTree - Hierarchy List. 2628
IVSetAsAvlTree - Member Functions and Data by Group 2628
IVSetAsAvlTree - Type Definitions 2629
IVSetAsAvlTree - Associated Globals 2629
IVSetAsAvlTree - Inherited Member Functions and Data 2629
IVSetAsBstTree 2631
IVSetAsBstTree - Hierarchy List 2631
IVSetAsBstTree - Member Functions and Data by Group 2632
IVSetAsBstTree - Type Definitions 2632
IVSetAsBstTree - Associated Globals 2632
IVSetAsBstTree - Inherited Member Functions and Data 2633
IVSetAsDilTable 2635
IVSetAsDilTable - Hierarchy List 2635
IVSetAsDilTable - Member Functions and Data by Group 2635
IVSetAsDilTable - Type Definitions. 2636
IVSetAsDilTable - Associated Globals 2636
IVSetAsDilTable - Inherited Member Functions and Data 2636
IVSetAsHshTable. 2638
IVSetAsHshTable - Hierarchy List 2638
IVSetAsHshTable - Member Functions and Data by Group 2639
IVSetAsHshTable - Type Definitions 2639
IVSetAsHshTable - Associated Globals . . . 2639
IVSetAsHshTable - Inherited Member Functions and Data 2640
IVSetAsList. 2642
IVSetAsList - Hierarchy List 2642
IVSetAsList - Member Functions and Data by Group 2642
IVSetAsList - Type Definitions 2643
IVSetAsList - Associated Globals 2643
IVSetAsList - Inherited Member Functions and Data 2643
IVSetAsTable 2645
IVSetAsTable - Hierarchy List 2645
IVSetAsTable - Member Functions and Data by Group 2646
IVSetAsTable - Type Definitions. 2646

IVSetAsTable - Associated Globals 2646
IVSetAsTable - Inherited Member Functions and Data 2647
IVSortedBag 2649
IVSortedBag - Hierarchy List 2649
IVSortedBag - Member Functions and Data by Group 2649
IVSortedBag - Type Definitions 2650
IVSortedBag - Associated Globals 2650
IVSortedBag - Inherited Member Functions and Data 2650
IVSortedBagAsAvlTree 2653
IVSortedBagAsAvlTree - Hierarchy List . . . 2653
IVSortedBagAsAvlTree - Member Functions and Data by Group 2653
IVSortedBagAsAvlTree - Type Definitions . . 2653
IVSortedBagAsAvlTree - Associated Globals 2654
IVSortedBagAsAvlTree - Inherited Member Functions and Data 2654
IVSortedBagAsBstTree 2657
IVSortedBagAsBstTree - Hierarchy List . . . 2657
IVSortedBagAsBstTree - Member Functions and Data by Group 2657
IVSortedBagAsBstTree - Type Definitions . . . 2657
IVSortedBagAsBstTree - Associated Globals 2658
IVSortedBagAsBstTree - Inherited Member Functions and Data 2658
IVSortedBagAsDilTable. 2660
IVSortedBagAsDilTable - Hierarchy List . . . 2660
IVSortedBagAsDilTable - Member Functions and Data by Group 2661
IVSortedBagAsDilTable - Type Definitions . . 2661
IVSortedBagAsDilTable - Associated Globals 2661
IVSortedBagAsDilTable - Inherited Member Functions and Data 2662
IVSortedBagAsList 2664
IVSortedBagAsList - Hierarchy List. 2664
IVSortedBagAsList - Member Functions and Data by Group 2665
IVSortedBagAsList - Type Definitions 2665
IVSortedBagAsList - Associated Globals . . . 2665
IVSortedBagAsList - Inherited Member Functions and Data 2666
IVSortedBagAsTable. 2668
IVSortedBagAsTable - Hierarchy List 2668
IVSortedBagAsTable - Member Functions and Data by Group 2669
IVSortedBagAsTable - Type Definitions . . . 2669
IVSortedBagAsTable - Associated Globals . . 2669
IVSortedBagAsTable - Inherited Member Functions and Data 2669
IVSortedMap 2672
IVSortedMap - Hierarchy List 2672
IVSortedMap - Member Functions and Data by Group 2673
IVSortedMap - Type Definitions. 2673
IVSortedMap - Associated Globals 2673
IVSortedMap - Inherited Member Functions and Data 2674
IVSortedMapAsAvlTree 2676
IVSortedMapAsAvlTree - Hierarchy List . . . 2676
IVSortedMapAsAvlTree - Member Functions and Data by Group 2677
IVSortedMapAsAvlTree - Type Definitions . . 2677
IVSortedMapAsAvlTree - Associated Globals 2677
IVSortedMapAsAvlTree - Inherited Member Functions and Data 2678
IVSortedMapAsBstTree. 2680
IVSortedMapAsBstTree - Hierarchy List . . . 2680
IVSortedMapAsBstTree - Member Functions and Data by Group 2681
IVSortedMapAsBstTree - Type Definitions . . 2681
IVSortedMapAsBstTree - Associated Globals 2681
IVSortedMapAsBstTree - Inherited Member Functions and Data 2682
IVSortedMapAsDilTable 2684
IVSortedMapAsDilTable - Hierarchy List . . . 2684
IVSortedMapAsDilTable - Member Functions and Data by Group 2685
IVSortedMapAsDilTable - Type Definitions 2685
IVSortedMapAsDilTable - Associated Globals 2685
IVSortedMapAsDilTable - Inherited Member Functions and Data 2686
IVSortedMapAsList 2688
IVSortedMapAsList - Hierarchy List 2688
IVSortedMapAsList - Member Functions and Data by Group 2689
IVSortedMapAsList - Type Definitions. . . . 2689
IVSortedMapAsList - Associated Globals . . . 2689
IVSortedMapAsList - Inherited Member Functions and Data 2689
IVSortedMapAsTable 2692
IVSortedMapAsTable - Hierarchy List 2692
IVSortedMapAsTable - Member Functions and Data by Group 2692
IVSortedMapAsTable - Type Definitions . . . 2693
IVSortedMapAsTable - Associated Globals 2693
IVSortedMapAsTable - Inherited Member Functions and Data 2693
IVSortedRelation 2696
IVSortedRelation - Hierarchy List 2696
IVSortedRelation - Member Functions and Data by Group 2696
IVSortedRelation - Type Definitions 2697
IVSortedRelation - Associated Globals. . . . 2697
IVSortedRelation - Inherited Member Functions and Data 2697
IVSortedRelationAsDilTable 2700
IVSortedRelationAsDilTable - Hierarchy List 2700
IVSortedRelationAsDilTable - Member Functions and Data by Group 2700
IVSortedRelationAsDilTable - Type Definitions 2701
IVSortedRelationAsDilTable - Associated Globals 2701
IVSortedRelationAsDilTable - Inherited Member Functions and Data 2701
IVSortedRelationAsTable 2704
IVSortedRelationAsTable - Hierarchy List . . 2704
IVSortedRelationAsTable - Member Functions and Data by Group 2704
IVSortedRelationAsTable - Type Definitions 2704
IVSortedRelationAsTable - Associated Globals 2705

IVSortedRelationAsTable - Inherited Member Functions and Data 2705
IVSortedSet 2708
IVSortedSet - Hierarchy List 2708
IVSortedSet - Member Functions and Data by Group 2708
IVSortedSet - Type Definitions 2708
IVSortedSet - Associated Globals 2709
IVSortedSet - Inherited Member Functions and Data 2709
IVSortedSetAsAvlTree 2711
IVSortedSetAsAvlTree - Hierarchy List . . . 2712
IVSortedSetAsAvlTree - Member Functions and Data by Group 2712
IVSortedSetAsAvlTree - Type Definitions . . . 2712
IVSortedSetAsAvlTree - Associated Globals 2712
IVSortedSetAsAvlTree - Inherited Member Functions and Data 2713
IVSortedSetAsBstTree 2715
IVSortedSetAsBstTree - Hierarchy List. . . . 2715
IVSortedSetAsBstTree - Member Functions and Data by Group 2716
IVSortedSetAsBstTree - Type Definitions . . . 2716
IVSortedSetAsBstTree - Associated Globals 2716
IVSortedSetAsBstTree - Inherited Member Functions and Data 2717
IVSortedSetAsDilTable 2719
IVSortedSetAsDilTable - Hierarchy List . . . 2719
IVSortedSetAsDilTable - Member Functions and Data by Group 2720
IVSortedSetAsDilTable - Type Definitions. . . 2720
IVSortedSetAsDilTable - Associated Globals 2720
IVSortedSetAsDilTable - Inherited Member Functions and Data 2720
IVSortedSetAsList 2723
IVSortedSetAsList - Hierarchy List 2723
IVSortedSetAsList - Member Functions and Data by Group 2723
IVSortedSetAsList - Type Definitions 2724
IVSortedSetAsList - Associated Globals . . . 2724
IVSortedSetAsList - Inherited Member Functions and Data 2724
IVSortedSetAsTable 2727
IVSortedSetAsTable - Hierarchy List 2727
IVSortedSetAsTable - Member Functions and Data by Group 2727
IVSortedSetAsTable - Type Definitions. . . . 2728
IVSortedSetAsTable - Associated Globals . . . 2728
IVSortedSetAsTable - Inherited Member Functions and Data 2728
IVStack 2731
IVStack - Hierarchy List 2731
IVStack - Member Functions and Data by Group 2731
IVStack - Type Definitions. 2731
IVStack - Associated Globals 2732
IVStack - Inherited Member Functions and Data 2732
IVStackAsDilTable 2734
IVStackAsDilTable - Hierarchy List. 2734
IVStackAsDilTable - Member Functions and Data by Group 2734
IVStackAsDilTable - Type Definitions 2735
IVStackAsDilTable - Associated Globals . . . 2735
IVStackAsDilTable - Inherited Member Functions and Data 2735
IVStackAsList 2737
IVStackAsList - Hierarchy List 2737
IVStackAsList - Member Functions and Data by Group 2737
IVStackAsList - Type Definitions 2738
IVStackAsList - Associated Globals. 2738
IVStackAsList - Inherited Member Functions and Data 2738
IVStackAsTable 2740
IVStackAsTable - Hierarchy List. 2740
IVStackAsTable - Member Functions and Data by Group 2741
IVStackAsTable - Type Definitions 2741
IVStackAsTable - Associated Globals 2741
IVStackAsTable - Inherited Member Functions and Data 2742

Chapter 7. Exceptions Signaled By Collection Classes 2745

IChildAlreadyExistsException 2745
IChildAlreadyExistsException - Hierarchy List 2745
IChildAlreadyExistsException - Member Functions and Data by Group 2745
IChildAlreadyExistsException - Inherited Member Functions and Data 2746
ICollectionLockException 2747
ICollectionLockException - Hierarchy List . . 2747
ICollectionLockException - Member Functions and Data by Group 2747
ICollectionLockException - Inherited Member Functions and Data 2749
ICollectionLockTimeOutException 2750
ICollectionLockTimeOutException - Hierarchy List 2750
ICollectionLockTimeOutException - Member Functions and Data by Group 2750
ICollectionLockTimeOutException - Inherited Member Functions and Data 2751
ICollectionResourceException 2752
ICollectionResourceException - Hierarchy List 2752
ICollectionResourceException - Member Functions and Data by Group 2752
ICollectionResourceException - Inherited Member Functions and Data 2754
ICollectionUnlockException 2754
ICollectionUnlockException - Hierarchy List 2754
ICollectionUnlockException - Member Functions and Data by Group 2755
ICollectionUnlockException - Inherited Member Functions and Data 2756
IEmptyException. 2757
IEmptyException - Hierarchy List 2757
IEmptyException - Member Functions and Data by Group 2757

IEmptyException - Inherited Member Functions and Data 2758
IFullException 2759
IFullException - Hierarchy List 2759
IFullException - Member Functions and Data by Group 2759
IFullException - Inherited Member Functions and Data 2761
IIdenticalCollectionException 2762
IIdenticalCollectionException - Hierarchy List 2762
IIdenticalCollectionException - Member Functions and Data by Group 2762
IIdenticalCollectionException - Inherited Member Functions and Data 2763
INotBoundedException 2764
INotBoundedException - Hierarchy List . . . 2764
INotBoundedException - Member Functions and Data by Group 2764
INotBoundedException - Inherited Member Functions and Data 2766
INotContainsKeyException 2767
INotContainsKeyException - Hierarchy List 2767
INotContainsKeyException - Member Functions and Data by Group 2767
INotContainsKeyException - Inherited Member Functions and Data 2768
IPositionInvalidException 2769
IPositionInvalidException - Hierarchy List . . 2769
IPositionInvalidException - Member Functions and Data by Group 2770
IPositionInvalidException - Inherited Member Functions and Data 2771
IPreconditionViolation 2772
IPreconditionViolation - Hierarchy List . . . 2772
IPreconditionViolation - Member Functions and Data by Group 2773
IPreconditionViolation - Inherited Member Functions and Data 2774
IRootAlreadyExistsException 2775
IRootAlreadyExistsException - Hierarchy List 2775
IRootAlreadyExistsException - Member Functions and Data by Group 2775
IRootAlreadyExistsException - Inherited Member Functions and Data 2776
IStreamerInvalidException 2777
IStreamerInvalidException - Hierarchy List 2777
IStreamerInvalidException - Member Functions and Data by Group 2777
IStreamerInvalidException - Inherited Member Functions and Data 2779
IStreamErrorException 2779
IStreamErrorException - Hierarchy List . . . 2780
IStreamErrorException - Member Functions and Data by Group 2780
IStreamErrorException - Inherited Member Functions and Data 2781

Chapter 8. Operations For Collection Classes 2783

IAMultiwayTreeOps 2783
IAMultiwayTreeOps - Hierarchy List 2783
IAMultiwayTreeOps - Member Functions and Data by Group 2783
IAMultiwayTreeOps - Inherited Member Functions and Data 2784
IATreeOps 2784
IATreeOps - Hierarchy List 2784
IATreeOps - Member Functions and Data by Group 2784
IATreeOps - Inherited Member Functions and Data 2784
ICOps 2784
ICOps - Hierarchy List 2785
ICOps - Member Functions and Data by Group 2785
ICOps - Inherited Member Functions and Data 2785
IECOps 2785
IECOps - Hierarchy List 2786
IECOps - Member Functions and Data by Group 2786
IECOps - Inherited Member Functions and Data 2786
IEHOps 2786
IEHOps - Hierarchy List 2786
IEHOps - Member Functions and Data by Group 2787
IEHOps - Inherited Member Functions and Data 2787
IEKCKeyOps 2787
IEKCKeyOps - Hierarchy List 2787
IEKCKeyOps - Member Functions and Data by Group 2788
IEKCKeyOps - Inherited Member Functions and Data 2788
IEKCOps 2788
IEKCOps - Hierarchy List 2788
IEKCOps - Member Functions and Data by Group 2788
IEKCOps - Inherited Member Functions and Data 2789
IEKEHKeyOps 2789
IEKEHKeyOps - Hierarchy List 2789
IEKEHKeyOps - Member Functions and Data by Group 2789
IEKEHKeyOps - Inherited Member Functions and Data 2789
IEKEHOps 2790
IEKEHOps - Hierarchy List 2790
IEKEHOps - Member Functions and Data by Group 2790
IEKEHOps - Inherited Member Functions and Data 2790
IEOps 2791
IEOps - Hierarchy List 2791
IEOps - Member Functions and Data by Group 2791
IEOps - Inherited Member Functions and Data 2791
IKCKeyOps 2792
IKCKeyOps - Hierarchy List 2792
IKCKeyOps - Member Functions and Data by Group 2792
IKCKeyOps - Inherited Member Functions and Data 2792
IKCOps 2792

IKCOps - Hierarchy List 2793
IKCOps - Member Functions and Data by Group 2793
IKCOps - Inherited Member Functions and Data 2793
IKEHKeyOps 2793
IKEHKeyOps - Hierarchy List 2794
IKEHKeyOps - Member Functions and Data by Group 2794
IKEHKeyOps - Inherited Member Functions and Data 2794
IKEHOps 2794
IKEHOps - Hierarchy List 2795
IKEHOps - Member Functions and Data by Group 2795
IKEHOps - Inherited Member Functions and Data 2795
IMultiwayTreeOps 2795
IMultiwayTreeOps - Hierarchy List 2795
IMultiwayTreeOps - Member Functions and Data by Group 2795
IMultiwayTreeOps - Inherited Member Functions and Data 2797
IStdAsOps 2797
IStdAsOps - Hierarchy List 2797
IStdAsOps - Member Functions and Data by Group 2798
IStdAsOps - Inherited Member Functions and Data 2798
IStdCmpOps 2798
IStdCmpOps - Hierarchy List. 2798
IStdCmpOps - Member Functions and Data by Group 2798
IStdCmpOps - Inherited Member Functions and Data 2799
IStdEqOps 2799
IStdEqOps - Hierarchy List 2799
IStdEqOps - Member Functions and Data by Group 2799
IStdEqOps - Inherited Member Functions and Data 2800
IStdHshOps 2800
IStdHshOps - Hierarchy List 2800
IStdHshOps - Member Functions and Data by Group 2800
IStdHshOps - Inherited Member Functions and Data 2800
IStdKeyOps 2801
IStdKeyOps - Hierarchy List 2801
IStdKeyOps - Member Functions and Data by Group 2801
IStdKeyOps - Inherited Member Functions and Data 2801
IStdMemOps 2802
IStdMemOps - Hierarchy List 2802
IStdMemOps - Member Functions and Data by Group 2802
IStdMemOps - Inherited Member Functions and Data 2802
IStdOps 2803
IStdOps - Hierarchy List 2803
IStdOps - Member Functions and Data by Group 2803
IStdOps - Inherited Member Functions and Data 2803
IWCOps. 2804
IWCOps - Hierarchy List 2804
IWCOps - Member Functions and Data by Group 2804
IWCOps - Inherited Member Functions and Data 2804
IWEHOps 2805
IWEHOps - Hierarchy List 2805
IWEHOps - Member Functions and Data by Group 2806
IWEHOps - Inherited Member Functions and Data 2806
IWEKCKeyOps 2806
IWEKCKeyOps - Hierarchy List 2807
IWEKCKeyOps - Member Functions and Data by Group 2807
IWEKCKeyOps - Inherited Member Functions and Data 2807
IWEKCOps. 2807
IWEKCOps - Hierarchy List 2808
IWEKCOps - Member Functions and Data by Group 2808
IWEKCOps - Inherited Member Functions and Data 2808
IWEKEHKeyOps 2809
IWEKEHKeyOps - Hierarchy List 2809
IWEKEHKeyOps - Member Functions and Data by Group 2809
IWEKEHKeyOps - Inherited Member Functions and Data 2809
IWEKEHOps 2810
IWEKEHOps - Hierarchy List 2810
IWEKEHOps - Member Functions and Data by Group 2810
IWEKEHOps - Inherited Member Functions and Data 2811
IWEOps. 2811
IWEOps - Hierarchy List 2812
IWEOps - Member Functions and Data by Group 2812
IWEOps - Inherited Member Functions and Data 2812
IWKCKeyOps 2812
IWKCKeyOps - Hierarchy List 2813
IWKCKeyOps - Member Functions and Data by Group 2813
IWKCKeyOps - Inherited Member Functions and Data 2813
IWKCOps 2813
IWKCOps - Hierarchy List 2814
IWKCOps - Member Functions and Data by Group 2814
IWKCOps - Inherited Member Functions and Data 2814
IWKEHKeyOps 2815
IWKEHKeyOps - Hierarchy List. 2815

IWKEHKeyOps - Member Functions and Data by Group 2815
IWKEHKeyOps - Inherited Member Functions and Data 2815
IWKEHOps 2816
IWKEHOps - Hierarchy List 2816
IWKEHOps - Member Functions and Data by Group 2816
IWKEHOps - Inherited Member Functions and Data 2817
IWStdAsOps 2817
IWStdAsOps - Hierarchy List. 2817
IWStdAsOps - Member Functions and Data by Group 2818
IWStdAsOps - Inherited Member Functions and Data 2818
IWStdCmpKeyOps 2818
IWStdCmpKeyOps - Hierarchy List 2819
IWStdCmpKeyOps - Member Functions and Data by Group 2819
IWStdCmpKeyOps - Inherited Member Functions and Data 2819
IWStdCmpOps 2819
IWStdCmpOps - Hierarchy List 2820
IWStdCmpOps - Member Functions and Data by Group 2820
IWStdCmpOps - Inherited Member Functions and Data 2820
IWStdEqKeyOps 2820
IWStdEqKeyOps - Hierarchy List 2821
IWStdEqKeyOps - Member Functions and Data by Group 2821
IWStdEqKeyOps - Inherited Member Functions and Data 2821
IWStdEqOps 2821
IWStdEqOps - Hierarchy List. 2822
IWStdEqOps - Member Functions and Data by Group 2822
IWStdEqOps - Inherited Member Functions and Data 2822
IWStdHshKeyOps 2822
IWStdHshKeyOps - Hierarchy List 2823
IWStdHshKeyOps - Member Functions and Data by Group 2823
IWStdHshKeyOps - Inherited Member Functions and Data 2823
IWStdHshOps. 2823
IWStdHshOps - Hierarchy List 2824
IWStdHshOps - Member Functions and Data by Group 2824
IWStdHshOps - Inherited Member Functions and Data 2824
IWStdKeyOps. 2824
IWStdKeyOps - Hierarchy List 2825
IWStdKeyOps - Member Functions and Data by Group 2825
IWStdKeyOps - Inherited Member Functions and Data 2825
IWStdMemOps 2825
IWStdMemOps - Hierarchy List 2826
IWStdMemOps - Member Functions and Data by Group 2826
IWStdMemOps - Inherited Member Functions and Data 2826
IWStdOps 2827
IWStdOps - Hierarchy List 2827
IWStdOps - Member Functions and Data by Group 2827
IWStdOps - Inherited Member Functions and Data 2828

Chapter 9. Math Classes 2829

c_exception. 2829
c_exception - Hierarchy List 2829
c_exception - Member Functions and Data by Group 2829
c_exception - Associated Globals 2829
c_exception - Inherited Member Functions and Data 2833
complex. 2834
complex - Hierarchy List 2834
complex - Member Functions and Data by Group 2834
complex - Associated Globals. 2836
complex - Inherited Member Functions and Data 2844
IBinaryCodedDecimal 2844
IBinaryCodedDecimal - Hierarchy List . . . 2844
IBinaryCodedDecimal - Member Functions and Data by Group 2844
IBinaryCodedDecimal - Inherited Member Functions and Data 2855
IDecimal 2855
IDecimal - Hierarchy List 2856
IDecimal - Member Functions and Data by Group 2856
IDecimal - Enumerations 2860
IDecimal - Inherited Member Functions and Data 2861
IDecimalBase 2861
IDecimalBase - Hierarchy List 2861
IDecimalBase - Member Functions and Data by Group 2861
IDecimalBase - Enumerations. 2864
IDecimalBase - Inherited Member Functions and Data 2865
IDecimalDataError 2865
IDecimalDataError - Hierarchy List. 2865
IDecimalDataError - Member Functions and Data by Group 2865
IDecimalDataError - Inherited Member Functions and Data 2867
IDecimalProxy 2867
IDecimalProxy - Hierarchy List 2867
IDecimalProxy - Member Functions and Data by Group 2868
IDecimalProxy - Inherited Member Functions and Data 2869
IDecimalResult 2870
IDecimalResult - Hierarchy List 2870

IDecimalResult - Member Functions and Data by Group 2870
IDecimalResult - Inherited Member Functions and Data 2872
IDecimalUtil 2873
IDecimalUtil - Hierarchy List 2873
IDecimalUtil - Member Functions and Data by Group 2873
IDecimalUtil - Associated Globals 2876
IDecimalUtil - Inherited Member Functions and Data 2878
ILongLong 2878
ILongLong - Hierarchy List 2878
ILongLong - Member Functions and Data by Group 2878
ILongLong - Inherited Member Functions and Data 2882
IPair 2882
IPair - Hierarchy List 2882
IPair - Member Functions and Data by Group 2882
IPair - Type Definitions 2887
IPair - Associated Globals 2887
IPair - Inherited Member Functions and Data 2889
IPoint 2889
IPoint - Hierarchy List 2889
IPoint - Member Functions and Data by Group 2889
IPoint - Inherited Member Functions and Data 2891
IPointArray 2891
IPointArray - Hierarchy List 2891
IPointArray - Member Functions and Data by Group 2892
IPointArray - Inherited Member Functions and Data 2894
IRange 2894
IRange - Hierarchy List. 2894
IRange - Member Functions and Data by Group 2894
IRange - Inherited Member Functions and Data 2895
IRectangle 2896
IRectangle - Hierarchy List 2896
IRectangle - Member Functions and Data by Group 2896
IRectangle - Type Definitions 2908
IRectangle - Associated Globals 2908
IRectangle - Inherited Member Functions and Data 2909
ISize 2909
ISize - Hierarchy List 2909
ISize - Member Functions and Data by Group 2909
ISize - Inherited Member Functions and Data 2911

Chapter 10. USL I/O Streaming Classes 2913

_CCSID_T 2913
_CCSID_T - Hierarchy List 2913
_CCSID_T - Member Functions and Data by Group 2913
_CCSID_T - Inherited Member Functions and Data 2913
filebuf 2913
filebuf - Hierarchy List 2914
filebuf - Member Functions and Data by Group 2914
filebuf - Inherited Member Functions and Data 2920
fstream 2921
fstream - Hierarchy List 2921
fstream - Member Functions and Data by Group 2921
fstream - Inherited Member Functions and Data 2926
fstreambase 2927
fstreambase - Hierarchy List 2927
fstreambase - Member Functions and Data by Group 2927
fstreambase - Inherited Member Functions and Data 2933
ifstream 2934
ifstream - Hierarchy List 2934
ifstream - Member Functions and Data by Group 2934
ifstream - Inherited Member Functions and Data 2939
ios 2940
ios - Hierarchy List 2940
ios - Member Functions and Data by Group 2940
ios - Enumerations 2950
ios - Inherited Member Functions and Data 2954
iostream. 2955
iostream - Hierarchy List 2955
iostream - Member Functions and Data by Group 2955
iostream - Inherited Member Functions and Data 2956
iostream_withassign. 2957
iostream_withassign - Hierarchy List 2957
iostream_withassign - Member Functions and Data by Group 2957
iostream_withassign - Inherited Member Functions and Data 2958
istream 2960
istream - Hierarchy List 2961
istream - Member Functions and Data by Group 2961
istream - Inherited Member Functions and Data 2985
istream_withassign 2986
istream_withassign - Hierarchy List 2986
istream_withassign - Member Functions and Data by Group 2986
istream_withassign - Inherited Member Functions and Data 2987
istrstream 2989
istrstream - Hierarchy List. 2989
istrstream - Member Functions and Data by Group 2989
istrstream - Inherited Member Functions and Data 2993
ofstream. 2994
ofstream - Hierarchy List 2995
ofstream - Member Functions and Data by Group 2995
ofstream - Inherited Member Functions and Data 2999

ostream 3000
ostream - Hierarchy List 3001
ostream - Member Functions and Data by Group 3001
ostream - Inherited Member Functions and Data 3014
ostream_withassign 3015
ostream_withassign - Hierarchy List 3015
ostream_withassign - Member Functions and Data by Group 3015
ostream_withassign - Inherited Member Functions and Data 3016
ostrstream 3018
ostrstream - Hierarchy List 3018
ostrstream - Member Functions and Data by Group 3018
ostrstream - Inherited Member Functions and Data 3021
stdiobuf 3022
stdiobuf - Hierarchy List 3022
stdiobuf - Member Functions and Data by Group 3022
stdiobuf - Inherited Member Functions and Data 3024
stdiostream. 3024
stdiostream - Hierarchy List 3025
stdiostream - Member Functions and Data by Group 3025
stdiostream - Inherited Member Functions and Data 3025
streambuf 3026
streambuf - Hierarchy List. 3027
streambuf - Member Functions and Data by Group 3027
streambuf - Inherited Member Functions and Data 3042
strstream 3042
strstream - Hierarchy List 3042
strstream - Member Functions and Data by Group 3042
strstream - Inherited Member Functions and Data 3045
strstreambase 3046
strstreambase - Hierarchy List 3046
strstreambase - Member Functions and Data by Group 3046
strstreambase - Inherited Member Functions and Data 3047
strstreambuf 3049
strstreambuf - Hierarchy List. 3049
strstreambuf - Member Functions and Data by Group 3049
strstreambuf - Inherited Member Functions and Data 3056

Chapter 11. Global Enums, Typedefs, and Variables 3057

Bibliography 3109

Notices 3111
Programming Interface Information 3112
Trademarks and Service Marks 3112
Industry Standards 3113

About This Reference

This reference describes the classes and class members of the C and C++ class libraries that are part of IBM Open Class™ Library, the comprehensive set of class libraries provided with the z/OS C/C++ product. The book covers the following IBM Open Classes:

- Application Control Classes
- Streams
- File Systems
- Text and International Classes
- Error Handling, Tracing, and Testing Classes
- Collection Classes
- Exceptions Signaled By Collection Classes
- Operations For Collection Classes
- Mathematics Classes
- Non-ANSI Input/Output Streaming Classes
- Global Enums, Typedefs, and Variables

Note: The PDF version of this publication is shipped as one volume. The BookManager® version is shipped as three separate volumes.

Who Should Use This Book

This book is intended for skilled C or C++ programmers who want to develop portable C or C++ applications using the IBM Open Class Library. Programmers using this book need to understand the concept of classes. For the Collection Class Library, programmers must also be familiar with using C/C++ templates. You should also know the operating system, and the related products as appropriate.

How to Use This Book

Classes are organized alphabetically within each class library, except where classes with similar uses or characteristics are grouped together. Functions and data members are listed alphabetically at the start of each class chapter, and their descriptions are grouped according to their purpose. If a class has more than one version of a function, all versions are described in one place.

A Note about Examples

Examples and code fragments that illustrate the use of the IBM Open Class Library use a simple style. They do not demonstrate all the uses of the IBM Open Class Library. Most examples and code fragments will not compile without additional code.

z/OS C/C++ and Related Publications

For a list of related z/OS C/C++ publications, see the "Bibliography" on page 3109.

© Copyright IBM Corp. 2001

How to Send Your Comments

Your feedback is important in helping to provide the most accurate and high-quality information. IBM welcomes any comments about this book or any other documentation.

- If you prefer to send comments by mail, use the the following address:

  IBM Canada Ltd. Laboratory
  Information Development
  B3/KB7/8200/MKM
  8200 Warden Avenue
  Markham, Ontario, Canada L6G 1C7

  If you are mailing a readers' comment form from a country other than the United States, you can give the form to the local IBM branch office or IBM representative for postage-paid mailing.
- If you prefer to send comments electronically, use one of these e-mail addresses:
  - Comments on books:

    torrcf@ca.ibm.com

    IBMLink: toribm(torrcf)

Be sure to include the following:

- The name of the book.
- The publication number of the book.
- The page number or topic to which your comment applies.

Chapter 1. Application Control Classes

IAttribute

The IAttribute class is an abstract base class for user defined attributes.

Class header file: iattribt.hpp

IAttribute - Hierarchy List

IAttribute

IAttribute - Member Functions and Data by Group

Constructors & Destructor

~IAttribute

```
public:virtual ~IAttribute()
```

This is supported on **WIN OS/2® AIX® OS/400® z/OS**

Destroys an object of this class.

Attribute Equality

Use these member functions to test attributes.

operator !=

```
public:bool operator !=(const IAttribute& anAttribute) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests two IAttributes for inequality.

operator ==

```
public:virtual bool
  operator ==( const IAttribute& anAttribute ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests two IAttributes for equality. This is an abstract member function.

Cloning Attributes

Use this member function to clone attributes.

clone

```
public:virtual IAttribute* clone() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a polymorphic copy of the attribute.

IAttribute - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

© Copyright IBM Corp. 2001

Inherited Protected Functions

None

Inherited Protected Data

None

IAttributeName

The IAttributeName class is a simple token class used to tag attributes.

Class header file: iatname.hpp

IAttributeName - Hierarchy List

IAttributeName

IAttributeName - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAttributeName

```
public:~IAttributeName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IAttributeName

Constructs an IAttributeName object.

Overload 1

```
public:IAttributeName(const char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IAttributeName(const IAttributeName&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment

Use this member to set attribute names.

operator =

```
public:IAttributeName& operator =(const IAttributeName& rhs)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Equality

Use these member functions to test attribute names.

operator !=

```
public:bool operator !=(const IAttributeName& rhs) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests two IAttributeNames for inequality.

operator ==

```
public:bool operator ==(const IAttributeName& rhs) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests two IAttributeNames for equality.

Query

Use these member funtions to get attribute names.

getString

```
public:const char* getString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the attribute name.

hash

```
public:ioc::unumeric_t hash() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the attribute name as a hash value for quick comparisons.

IAttributeName - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICondition

ICondition is the abstract base class for all condition classes. Conditions are used in two ways. They can provide monitor/condition style services and they can provide traditional event semaphore services.

Threads can wait on conditions (using the wait() method), and other threads can signal or broadcast the condition to let waiting threads continue execution. This is the standard event semaphore paradigm. The condition represents an 'auto mode' event, that either lets one waiting thread go (signal) or lets all waiting threads go (broadcast.) Once the signal or broadcast completes, it resets itself automatically (as apposed to a manual mode event that stays triggered until it is manually reset.)

To also support monitor/condition type of services, there is an alternative form of the wait() method that takes a resource lock. While the thread is in the condition wait, the resource lock is unlocked. This lets multiple threads fight over a single resource while simultaneously being blocked periodically on a condition. The resource object that is locked by the passed resource lock is unwound as many times as necessary to unlock the resource. When the thread wakes up from the condition wait then it will continue to block until it can regain the resource lock.

Once regained, the resource is wound back up as many times as it previously was locked. Since all recursive locks must occur on the same thread, this is safe.

Class header file: iconditn.hpp

ICondition - Hierarchy List

ICondition

ISharedCondition

IPrivateCondition

ICondition - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICondition objects.

~ICondition

```
public:virtual ~ICondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor has nothing to do at this level and is just an empty inline implementation.

ICondition

```
protected:ICondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ICondition constructors are not publically available since it is an abstract base class. The constructors that are implemented are only available to derived classes or friends.

Locking Methods

Use these member functions to unlock and relock IResourceLock objects.

Relock

```
public:static void Relock(IResourceLock&, ioc::numeric_t times)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Relocks the IResourceLock the specified number of times.

Unlock

```
public:static ioc::numeric_t Unlock(IResourceLock&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Unlocks the IResourceLock.

Synchronization Methods

Use these member functions to define the synchronization interface for conditions.

broadcast

```
public:virtual void broadcast() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows all threads, currently waiting on this condition, to return from their wait operations.

In some platform implementations, threads entering wait() while the broadcast is being done might also be allowed to pass on through (it does not necessarily take a snapshot of waiting threads).

Once there are no more threads to let go, the condition is reset. It does not stay signaled.

Derived classes should override this method and implement the broadcast as required for the platform.

signal

```
public:virtual void signal() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Signals the condition, which allows one other thread to return from a wait on the condition.

Only one thread is allowed through. If no threads are waiting, then this has no effect. Therefore, signal() is pulse type operation. It does not remain signaled.

Derived classes must override this method and implement the signal operation as required for the platform.

wait

Waits for the condition to be signaled or broadcast by another thread or process. An optional timeout can be provided, which defaults to an infinite wait if none is explicitly given.

Derived classes should put the calling thread to sleep until another thread does a signal() or broadcast() call, or until the specified timeout expires (if a signal or broadcast does not arrive within the time out period).

Overload 1

```
public:virtual bool wait(ioc::numeric_t timeOut = - 1) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool
  wait( IResourceLock& alock,
        ioc::numeric_t timeOut = - 1 ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ICondition - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICountedPointerTo

Creates a templatized smart pointer with automatic reference counting.

ICountedPointerTo provides a smart C++ like pointer with this additional built-in behavior: when you create an instance of ICountedPointerTo, it automatically calls IMRefCounted::addRef on the object pointed to; when the ICountedPointerTo is destroyed, it calls IMRefCounted::removeRef to remove its reference to the object.

ICountedPointerTo also allows you to perform reference counting on objects that do not descend from IMRefCounted. In this case, it encloses, or wraps, the object in a reference-counted wrapper.

Class header file: icntptr.hpp

ICountedPointerTo - Hierarchy List

ICountedPointerTo

ICountedPointerTo - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct counted pointer objects.

~ICountedPointerTo

```
public:~ICountedPointerTo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the counted pointer and decrements the reference count of the object pointed to. When the count is zero, the object is deleted.

ICountedPointerTo

Constructs an object of class ICountedPointerTo.

Overload 1

```
public:ICountedPointerTo(const ICountedPointerTo < AType >& share)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a copy of the given counted pointer. The constructor increments the reference count of the object pointed to by the given counted pointer.

Overload 2

```
public:ICountedPointerTo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an object that points to nothing. This function is slightly more efficient than calling ICountedPointerTo(0).

Overload 3

```
public:ICountedPointerTo(AType* adopt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an object of class ICountedPointerTo that points to the object specified by the adopt parameter. This constructor adopts the given object and increments its reference count. Use the newly constructed instance of ICountedPointer to make any further references to the object pointed to. Discard the raw pointer used to specify the adopted object.

adopt A pointer to the object to be pointed to from the constructed object of class ICountedPointerTo.

Accessing an Object

Use these member functions to access the pointer being counted.

getAlias

```
public:AType* getAlias() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a raw C++ pointer to the object to which this points. If this pointer does not refer to an object, the function returns zero.

Use this function to pass a raw C++ pointer to an existing function that expects one. Do not continue to use the pointer after this ICountedPointerTo object has been destroyed.

operator *

```
public:AType& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator returns a reference to the object referred to by the counted pointer.

Exception

Exception	Description
IInvalidRequest	Thrown when your code attempts to dereference a null pointer.

operator ->

```
public:AType* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is called when you use a counted pointer with the C++ class member access operator ->.

Exception

Exception	Description
IInvalidRequest	Thrown when your code attempts to dereference a null pointer.

Changing the Object Pointed to

Use these member functions to point this counted pointer object at another object.

operator =

This operator modifies the counted pointer so that it refers to a different object from the one it currently points to.

Overload 1

```
public:ICountedPointerTo < AType >&
  operator =( const ICountedPointerTo < AType >& share )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to modify the counted pointer so that it refers to the same object as another counted pointer. If the object the pointer previously referred to was a valid object, that object's reference count is decremented. When the count is zero, that object is deleted.

share A counted pointer to the new target object.

Overload 2

```
public:ICountedPointerTo < AType >& operator =(AType* adopt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to modify the counted pointer so that it refers to the given object to be adopted. If the object the pointer previously referred to was a valid object, that object's reference count is decremented. The object is deleted when its reference count is zero.

The function adopts the given object and increments its reference count. Make all further references to the given object through this ICountedPointer object. Discard the raw pointer used to specify the adopted object.

adopt The object to be pointed to. The operator assigns the adopted object to the instance of ICountedPointerTo and increments the object's reference count.

Comparing and Testing the Pointer

Use these member functions to determine whether this object points to a particular object or the same object that another counted pointer object also points to, and whether the pointer is valid or not.

operator !=

This operator determines whether the counted pointer refers to a particular object.

Overload 1

```
public:bool operator !=(const AType* r) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to determine if this counted pointer refers to the given object.

Overload 2

```
public:bool
  operator !=( const ICountedPointerTo < AType >& r ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to determine if this counted pointer refers to the same object as another counted pointer.

operator ==

This operator determines whether this pointer refers to a particular object.

Overload 1

```
public:bool
  operator ==( const ICountedPointerTo < AType >& r ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to determine if this counted pointer refers to the same object as another counted pointer.

Overload 2

```
public:bool operator ==(const AType* r) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to determine if this counted pointer refers to the given object.

valid

```
public:bool valid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Queries whether this counted pointer refers to a valid object. Returns true if this counted pointer refers to a valid, non-null object. Otherwise it returns false.

validate

```
public:void validate() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Asserts that this pointer refers to a valid object. If it does not, the function throws an exception.

Exception

Exception	Description
IInvalidRequest	This pointer does not refer to a valid object.

Counting Object References

Use these member functions to manage the reference count of an object.

count

```
public:ioc::unumeric_t count() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the reference count of the object to which this counted pointer refers. If this counted pointer does not refer to a valid object, it returns zero.

Note that when the counted pointer refers to an object that does not inherit from IMRefCounted, this function returns the reference count of the internal wrapper around that object.

ICountedPointerTo - Inherited Member Functions and Data

Inherited Public Functions

ICountedPointerTo			
Definition	**Page Number**	**Definition**	**Page Number**
`getCounted` `isWrapped`	6 6	`in`	6

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICritSec

The ICritSec class isolates blocks of code you consider to be critical sections. Such blocks usually manipulate static data and need to do so without interruption from code that is executing in other threads.

Objects of this class guarantee that no other thread in the current process executes between the object's construction and destruction.

Typically, code that must execute without interruption by other threads is partitioned into a separate block, delimited by braces ({}). You create an object of this class within that block.

Note: While the critical section is executing, avoid actions having the potential to interrupt it, such as calls to the operating system or Open Class Library. If you must preserve locks beyond the scope of a critical section object, use semaphores as implemented by the class IResourceLock.

AIX Considerations

Use this class to temporarily block signals from the current process. The constructor sets the process signal mask to block all maskable signals except SIGTRAP. SIGTRAP is not masked to allow debug tracing through a critical section. The destructor restores the process signal mask to the value it had before creation of the object.

Because signals are masked, some services might not operate correctly inside a critical section. An example of this is timer signals. If you attempt to lock an ISharedResource inside a critical section, any timeout you specify will be ineffective because the timer signal is masked.

Class header file: icritsec.hpp

ICritSec - Hierarchy List

ICritSec

ICritSec - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class. The constructor and destructor implement all of the function of this class. The constructor causes a critical section to be entered. The corresponding destructor exits the critical section.

~ICritSec

```
public:~ICritSec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Exits a critical section within the block of code where you create the critical section object and deletes the object.

ICritSec

```
public:ICritSec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a critical section object and enters a critical section within the block of code where you create the object.

Diagnostics

Use these member functions for diagnostic purposes. They report additional information about critical sections.

count

```
public:static unsigned long count()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the total number of critical sections entered but not yet exited. It is equivalent to the number of ICritSec objects in existence.

ICritSec - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICrossThreadNotificationLoop

ICrossThreadNotificationLoop starts the asynchronous notification dispatch loop for non-GUI threads. A thread which is not running processMsgs is a non-GUI thread. This loop is required if you are creating observers which are interested in receiving asynchronous notifications in a non-GUI thread.

Class header file: iobservr.hpp

ICrossThreadNotificationLoop - Hierarchy List

ICrossThreadNotificationLoop

ICrossThreadNotificationLoop - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destroy objects of this class. Only construct or destruct one object of this class per thread.

~ICrossThreadNotificationLoop

```
public:virtual ~ICrossThreadNotificationLoop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys this object.

ICrossThreadNotificationLoop

```
public:ICrossThreadNotificationLoop(IObserver&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICrossThreadNotificationLoop object.

Creates the object using an observer that was created in a non-GUI thread. The loop uses the observer to get to the asynchronous request processor in order to process notifications.

Notification Processing

Use these member functions to affect the dispatching of asynchronous notification events in threads which are not running with processMsgs.

cleanup

```
protected:void cleanup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called before the run method exits.

dispatchNotification

```
protected:virtual void dispatchNotification()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Dispatches the next pending notification for the thread. If there are no notifications to process, it returns immediately.

observersInCurrentThread

```
protected:bool observersInCurrentThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if there are observers registered to receive notifications in this thread.

waitAndDispatchNotification

```
protected:virtual void waitAndDispatchNotification()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Dispatches the next pending notification for the thread. If there are no notifications to process, it waits.

Starting and Stopping

Use these member functions to start and stop the processing of notification events.

run

```
public:virtual void run()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts processing asynchronous notifications.

stop

```
public:virtual void stop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the loop from processing asynchronous notifications.

ICrossThreadNotificationLoop - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICurrentNonGUIApplication

This class provides access to the current application. There is a certain amount of functionality which does not require any access to the INonGUIApplication object, so it can be accessed via this class for greater efficiency.

Class header file: ingapp.hpp

ICurrentNonGUIApplication - Hierarchy List

INonGUIApplication

ICurrentNonGUIApplication

ICurrentNonGUIApplication - Member Functions and Data by Group

Constructors & Destructor

The constructor and destructor for this class are protected. You must derive from this class if you want to use it to represent a current non-GUI application.

~ICurrentNonGUIApplication

```
protected:virtual ~ICurrentNonGUIApplication()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys this object. Does not affect the process if it is still running.

ICurrentNonGUIApplication

```
protected:ICurrentNonGUIApplication()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICurrentNonGUIApplication object.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a string of formatted debug information for this object.

Process Arguments

Use these member functions to set and access the process' command line arguments.

argc

```
public:virtual int argc() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the count of arguments for this process.

argv

```
public:virtual IString argv(int argumentNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a particular argument for this process. The passed argument number indicates which one. Argument index begins at 0.

argvW

```
public:virtual IText argvW(int argumentNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a particular argument for this process as an IText object. The passed argument number indicates which one. Argument index begins at 0.

setArgs

```
public:virtual ICurrentNonGUIApplication&
  setArgs( int argc,
           const char * const argv [ ] )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the arguments for this process. It will not affect the process if the process is already started.

setArgsW

```
public:virtual ICurrentNonGUIApplication&
  setArgsW( int argc,
            const ioc::unichar_t * const argv [ ] )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the arguments for this process with Unicode strings. It will not affect the process if the process is already started.

Process Control

Use these member functions to control the current process.

exit

```
public:virtual ICurrentNonGUIApplication& exit()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Causes the current application to exit.

Process Information

Use these member functions to access process information.

pib

```
protected:struct pib_s& pib()
```

This is supported on **OS/2**

Return a reference to an OS/2 process information block (PIB) structure.

ICurrentNonGUIApplication - Inherited Member Functions and Data

Inherited Public Functions

INonGUIApplication			
Definition	**Page Number**	**Definition**	**Page Number**
adjustPriority	59	asString	59
current	59	currentPID	59
id	59	setPriority	59
setProcessPriority	60		

Inherited Public Data

None

Inherited Protected Functions

INonGUIApplication			
Definition	**Page Number**	**Definition**	**Page Number**
~INonGUIApplication	58	INonGUIApplication	58
setId	58		

Inherited Protected Data

None

ICurrentNonGUIThread

The ICurrentNonGUIThread class represents the current thread of execution. An object of this class contains information that the Open Class Library maintains for the current thread of execution. You are limited to a single object of this class.

This class provides functions that you can only apply to the current thread of execution. To obtain a reference to the object, use the static function INonGUIThread::current.

Class header file: ingthrd.hpp

ICurrentNonGUIThread - Hierarchy List

INonGUIThread

ICurrentNonGUIThread

ICurrentNonGUIThread - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~ICurrentNonGUIThread

```
public:virtual ~ICurrentNonGUIThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the object.

ICurrentNonGUIThread

```
public:ICurrentNonGUIThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICurrentNonGUIThread object.

Current Thread Information

Use these member functions to access general information on the current thread.

handle

```
public:virtual IThreadHandle handle() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the thread handle for the current thread.

OS/2 Considerations

The thread handle and thread ID are identical.

Windows Considerations

You must use the thread handle for the operating system thread and synchronization functions.

id

```
public:virtual IThreadId id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current thread's identifier (thread id).

Current Thread Support

Use these member functions to perform thread operations that only apply to the current thread.

remainingStack

```
public:virtual ioc::unumeric_t remainingStack() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an approximation of the number of available bytes that remain on the stack. If the stack is not initialized, 0 is returned.

Windows Considerations

This function always returns 1000 because an approximation cannot be obtained.

waitFor

Allows the current thread to wait for a specified thread to end.

Overload 1

```
public:virtual ICurrentNonGUIThread&
  waitFor( const INonGUIThread& anotherThread )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Waits for another thread without any timeout. It waits forever.

anotherThread

The thread to wait for.

Overload 2

```
public:virtual ICurrentNonGUIThread&
  waitFor( const INonGUIThread& anotherThread,
           ioc::numeric_t timeout,
           bool* timedOut )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:virtual ICurrentNonGUIThread&
  waitFor( const INonGUIThread& anotherThread,
           ioc::numeric_t timeout )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

waitForAllThreads

```
public:virtual ICurrentNonGUIThread&
  waitForAllThreads( ioc::numeric_t timeout = - 1,
                     bool* timedOut = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows the main thread to wait for all other threads that were started using INonGUIThread or IThread to end. This function may only be called from the main thread of the process.

timeout

-1 means wait indefinitely, 0 means do not wait, other positive values give the timeout period in milliseconds.

timedOut

An optional boolean value that is set to indicate whether the return was because of a timeout or a successful wait.

waitForAnyThread

```
public:virtual IThreadId
  waitForAnyThread( ioc::numeric_t timeout = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows the current thread to wait for the first termination to occur of any other threads in the current process. It accepts a timeout value, in milliseconds, that indicates how long to wait before giving up. The timeout defaults to -1, which means wait forever. This function returns the ID of the thread that ended.

timeout

The number of milliseconds to wait for any other thread to end.

Starting and Stopping Threads

Use these member functions to control the starting and stopping of threads.

exit

```
public:virtual void exit(ioc::unumeric_t returnCode)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Ends the current thread with the specified return value.

returnCode

The code to return as the exit code for the thread.

sleep

```
public:virtual ICurrentNonGUIThread&
  sleep( ioc::unumeric_t milliseconds )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Suspends the thread for a specified number of milliseconds.

milliSeconds

The number of milliseconds to sleep.

suspend

```
public:virtual void suspend()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Suspends the current thread of execution.

ICurrentNonGUIThread - Inherited Member Functions and Data

Inherited Public Functions

INonGUIThread			
Definition	**Page Number**	**Definition**	**Page Number**
~INonGUIThread	61	adjustPriority	63
asDebugInfo	65	asString	65
convertedPriority	64	current	70
currentHandle	70	currentId	70
defaultStackSize	66	exitCode	70
INonGUIThread	61	isAutoDetachThread	65
isStarted	71	priorityClass	64
priorityLevel	64	resume	67
setAutoDetachThread	65	setDefaultStackSize	66
setPriority	64	setStackSize	66
setThreadPriority	71	setVariable	71

INonGUIThread			
Definition	**Page Number**	**Definition**	**Page Number**
stackSize stop variable	66 69 71	start threadPriority	67 72

Inherited Public Data

None

Inherited Protected Functions

INonGUIThread			
Definition	**Page Number**	**Definition**	**Page Number**
newStartedThread	69	operator =	61

ICurrentNonGUIThread			
Definition	**Page Number**	**Definition**	**Page Number**
startedThread	16		

Inherited Protected Data

None

IDLLModule

The IDLLModule supports the loading, unloading, and addressing of entry points in a dynamic link library (DLL). Dynamic link libraries are known as shared libraries or service programs on some platforms. IDLLModule makes it easier and safer to track loaded DLLs. It provides access to module information for those loaded DLLs, such as entry point addresses and the file name of the module.

Class header file: idllmod.hpp

IDLLModule - Hierarchy List

IDLLModule

IDLLModule - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct, copy, and assign objects of this class.

~IDLLModule

```
public:~IDLLModule()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The dynamic link library is unloaded from memory if it is no longer being referenced. The IDLLModule object is destroyed.

IDLLModule

Constructs an IDLLModule object.

Overload 1

```
public:IDLLModule(const IModuleHandle& moduleHandle)
```

This is supported on **WIN OS/2 OS/400**

Use this version of the constructor if you have an IModuleHandle obtained by previously loading the dynamic link library (DLL or service program).

moduleHandle
Reference to a module handle of a dynamic link library that has been loaded explicitly.

Exception

Exception	Description
IAccessError	The dynamic link library object was not created. The module handle may be invalid.

Overload 2

```
public:IDLLModule( const IText& fileName,
            ESearchLocation searchOrder = kOSDefaultSearch )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the constructor if you know the name of the dynamic link library (DLL, shared library, or service program). This constructor opens the dynamic link library.

Note: If you want to search the library path (LIBPATH or PATH) for the dynamic link library, specify **fileName** without the path. If no file extension is supplied, the OS specific default file extension will be used.

fileName
The dynamic link library name.

searchLocation
Search mechanism to use when the dynamic link library is not path qualified. No searching is performed on path qualified names.

OS/400 Considerations

On OS/400 the *LIBL is searched for the service program.

Overload 3

```
public:IDLLModule(const IDLLModule& dllLibrary)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a dynamic link library object using a reference to an existing dynamic link library object.

dllLibrary
Reference to an existing IDLLModule object.

operator =

```
public:IDLLModule& operator =(const IDLLModule& dllLibrary)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the member data of an object of this class to another object of this class.

Dynamic Link Library Information

Use these member functions to query information about the loaded dynamic link library.

fileName

```
public:IString fileName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the fully qualified file name of the dynamic link library.

Exception

Exception	Description
IAccessError	The file name was not returned. The module handle may be invalid.

fileNameW

```
public:IText fileNameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the fully qualified file name of the dynamic link library as an IText object.

handle

```
public:IModuleHandle handle() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the module handle of the dynamic link library.

isOpen

```
public:bool isOpen() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the dynamic link library is open.

Opening and Closing

Use these member functions to open (load) and close (unload) a dynamic link library.

close

```
public:IDLLModule& close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Decrements the reference count of the dynamic link library. If the count reaches 0, this method also closes the library.

open

```
public:IDLLModule& open()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Opens a closed dynamic link library and increments the reference count.
Exception

Exception	Description
IAccessError	The dynamic link library object was not created. The system either could not locate the file, or (for OS/2) the LIBRARY statement in the .def file does not match the DLL name.

Procedure Address Loading

Use these member functions to call a function in a dynamic link library. You can identify a function by its name or ordinal number.
isEntryPoint32Bit

Returns true if the entry point is 32-bit.

Overload 1

```
public:bool
  isEntryPoint32Bit( ioc::unumeric_t procedureOrdinal ) const
```

This is supported on **WIN OS/2 OS/400**

procedureOrdinal
Unsigned long value that represents the procedure ordinal.

Use this version of the function if you know the entry point's ordinal.
Exception

Exception	Description
IAccessError	The entry point was not queried. The procedure ordinal may be invalid.

OS/400 Considerations

On OS/400 this function will always return true.
OS/2 Considerations

If the entry point is 16-bit, this function will return false.
Windows Considerations

Only 32-bit functions are supported on Windows. This function will always return true.

Overload 2

```
public:bool isEntryPoint32Bit(const char* procedureName) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the function if you know the entry point's name.

procedureName
Pointer to the procedure name.
Exception

Exception	Description
IAccessError	The entry point was not queried. The procedure name may be invalid.

AIX Considerations

If the entry point is 64-bit, this function will return false.

OS/400 Considerations

On OS/400 this function will always return true.

OS/2 Considerations

If the entry point is 16-bit, this function will return false.

Windows Considerations

Only 32-bit functions are supported on Windows. This function will always return true.

procAddress

Returns the address of the procedure.

Overload 1

```
public:void* procAddress(ioc::unumeric_t procedureOrdinal) const
```

This is supported on **WIN OS/2 OS/400**

Use this version of the function if you know the procedure ordinal.

procedureOrdinal
Unsigned long value that represents the procedure ordinal.

Exception

Exception	Description
IAccessError	The procedure address was not loaded. The procedure ordinal may be invalid.

Overload 2

```
public:void* procAddress(const char* procedureName) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the function if you know the procedure name.

procedureName
Pointer to the procedure name.

Exception

Exception	Description
IAccessError	The procedure address was not loaded. The procedure name may be invalid.

IDLLModule - Enumerations

ESearchLocation

Use these enumerators to specify how to search for a dynamic link library (DLL or shared library). Valid values are:

kOSDefaultSearch
Finds a dynamic link library using the PATH or LIBPATH environment variables following the search rules of the native operating system.

kNLSPathSearch
Finds the dynamic link library using the NLSPATH environment variable after substituting %L with the user's current locale. (On Windows and OS/2, it is the LANG environment variable.)

AIX Considerations
Instead of using the LANG environment, the current user's locale is retrieved from a call to setlocale(LC_MESSAGES, NULL).

IDLLModule - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IDynamicLinkLibrary

The IDynamicLinkLibrary class supports the loading of resources from a dynamic link library (DLL). Dynamic link libraries are known as shared libraries on some platforms.

Class header file: ireslib.hpp

IDynamicLinkLibrary - Hierarchy List

IResourceLibrary

IDynamicLinkLibrary

IDynamicLinkLibrary - Member Functions and Data by Group

Constructors & Destructor

You can construct, copy, assign, and destruct objects of this class. You can construct objects of this class using either a dynamic link library name, module handle, or a reference to an existing object of this class.

~IDynamicLinkLibrary

```
public:virtual ~IDynamicLinkLibrary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The dynamic link library is unloaded from memory (closed) if it is no longer being referenced. Destructs the IDynamicLinkLibrary object.

IDynamicLinkLibrary

Constructs an IDynamicLinkLibrary object.

Overload 1

```
public:IDynamicLinkLibrary( const char* fileName,
                    ESearchLocation searchLocation = kOSDefaultSearch )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the constructor if you know the name of the dynamic link library (DLL or shared library). This constructor opens the dynamic link library.

Note: If you want to search the library path (LIBPATH or PATH) for the dynamic link library, specify **fileName** without the path or extension.

fileName
The dynamic link library name.

searchLocation
Search mechanism to use when the dynamic link library is not path qualified. No searching is performed on path qualified names.

Exception

Exception	Description
IAccessError	The dynamic link library object was not created. The system either could not locate the file, or (for OS/2) the LIBRARY statement in the .def file used to build the DLL does not match the DLL name. You also may receive this error if you used the **rename** system command to rename the dynamic link library rather than the utility program DLLRNAME.

Overload 2

```
public:IDynamicLinkLibrary(const IDynamicLinkLibrary& dllLibrary)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a dynamic link library object using a reference to an existing dynamic link library object.

dllLibrary
Reference to an existing IDynamicLinkLibrary object.

Overload 3

```
public:IDynamicLinkLibrary(const IModuleHandle& moduleHandle)
```

This is supported on **WIN OS/2 OS/400 z/OS**

Use this version of the constructor if you have an IModuleHandle obtained by loading the dynamic link library (DLL) explicitly.

moduleHandle
Reference to a module handle of a DLL that has been loaded explicitly.

Exception

Exception	Description
IAccessError	The dynamic link library object was not created. The module handle may be invalid.

operator =

```
public:IDynamicLinkLibrary&
  operator =( const IDynamicLinkLibrary& dllLibrary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the member data of an object of this class to another object of this class.

dllLibrary
Reference to an existing dynamic link library object.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides textual information about the class object. It returns a string that contains the file name of the dynamic link library.

asString

```
public:virtual IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides textual information about the class object. It returns a string that contains the file name of the dynamic link library.

Dynamic Link Library Information

Use these member functions to query general dynamic link library information.

fileName

```
public:virtual IString fileName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the fully qualified file name of the dynamic link library.
Exception

Exception	Description
IAccessError	The file name was not returned. The module handle may be invalid. Check to ensure that the dynamic link library has been properly opened.

handle

```
public:virtual IModuleHandle handle() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the module handle of the dynamic link library. If the dynamic link library is not open, the module handle is 0.

isOpen

```
public:virtual bool isOpen() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the dynamic link library is open.

resourceAddress

```
protected:virtual void* resourceAddress() const
```

This is supported on **AIX**

This is an internal use only function used to load resources on AIX.

Opening and Closing

Use these member functions to open (load) or close (unload) a dynamic link library (DLL).

Note: DLL objects are reference-counted. When a DLL is opened, the reference count is incremented, and when it is closed, the reference count is decremented. When the reference count reaches 0, the DLL is unloaded from memory.

close

```
public:virtual IDynamicLinkLibrary& close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Decrements the reference count of the dynamic link library. If the reference count reaches 0, this method also closes the library.

open

```
public:virtual IDynamicLinkLibrary& open()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Opens (loads) a closed dynamic link library and increments the reference count.

Exception

Exception	Description
IAccessError	The dynamic link library object was not created. The system either could not locate the file, or (for OS/2) the LIBRARY statement in the .def file does not match the DLL name. You also may receive this error if you used the **rename** system command to rename the dynamic link library rather than the utility program DLLRNAME.

Procedure Address Loading

Use these member functions to get a procedure address from a dynamic link library or to test the memory model of an entry point.

isEntryPoint32Bit

If the entry point is a 32-bit function, true is returned.

Overload 1

```
public:bool
  isEntryPoint32Bit( unsigned long procedureOrdinal ) const
```

This is supported on **WIN OS/2 OS/400 z/OS**

Use this version of the function if you know the entry point's ordinal.

procedureOrdinal
Unsigned long value that represents the procedure ordinal.

Exception

Exception	Description
IAccessError	The entry point was not queried. The procedure ordinal may be invalid or the dynamic link library may not be open.

Overload 2

```
public:bool isEntryPoint32Bit(const char* procedureName) const
```

This is supported on **WIN OS/2 OS/400 z/OS**

Use this version of the function if you know the entry point's name.

procedureName
Pointer to the procedure name.

Exception

Exception	Description
IAccessError	The entry point was not queried. The procedure name may be invalid or the dynamic link library may not be open.

procAddress

Returns the address of the specified procedure address from a dynamic link library.

Overload 1

```
public:void* procAddress(unsigned long procedureOrdinal) const
```

This is supported on **WIN OS/2 OS/400 z/OS**

Use this version of the function if you know the procedure ordinal.

procedureOrdinal
Unsigned long value that represents the procedure ordinal.

Exception

Exception	Description
IAccessError	The procedure address was not loaded. The procedure ordinal may be invalid or the dynamic link library may not be open.

Overload 2

```
public:void* procAddress(const char* procedureName) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the function if you know the procedure name.

procedureName
Pointer to the procedure name.

Exception

Exception	Description
IAccessError	The procedure address was not loaded. The procedure name may be invalid or the dynamic link library may not be open.

IDynamicLinkLibrary - Enumerations

ESearchLocation

Use these enumerators to specify how to search for a dynamic link library (DLL or shared library). Valid values are:

kOSDefaultSearch
Finds a dynamic link library using the PATH or LIBPATH environment variables following the search rules of the native operating system.

kNLSPathSearch
Finds the dynamic link library using the NLSPATH environment variable after substituting %L with the users' current locale. (On Windows and OS/2, it is the LANG environment variable.)

AIX Considerations
Instead of using the LANG environment, the user's current locale is retrieved from a call to setlocal(LC_MESSAGES, NULL).

IDynamicLinkLibrary - Inherited Member Functions and Data

Inherited Public Functions

IResourceLibrary			
Definition	**Page Number**	**Definition**	**Page Number**
~IResourceLibrary	115	IResourceLibrary	115
loadAccelTable	117	loadBitmap	117
loadDialog	118	loadHelpTable	119
loadIcon	119	loadMenu	120
loadMessage	120	loadPointer	121
loadString	121	operator =	116
tryToLoadBitmap	122	tryToLoadIcon	122
tryToLoadMessage	123	tryToLoadPointer	123
tryToLoadString	123		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEnumHandle

This class represents an enumeration handle.

Class header file: ibhandle.hpp

IEnumHandle - Hierarchy List

IEnumHandle

IEnumHandle - Member Functions and Data by Group

Constructors & Destructor

Use this member function to construct an IEnumHandle object.

IEnumHandle

```
public:IEnumHandle(Value henum = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IEnumHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

Diagnostics

Functions for accessing the handle's value.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

Type Conversions

Use these type conversion operators to convert objects of this class to a different type.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

IEnumHandle - Type Definitions

Value

A typedef for the host specific type of the enum handle.

Variation 1

```
typedef unsigned long Value
```

This is supported on **OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef void * Value
```

This is supported on **WIN**

IEnumHandle - Associated Globals

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IEnumHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IEnumHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEnumHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEnvironment

IEnvironment is a mechanism for building and manipulating environmental variables for passing to the IExternalProcess class. IEnvironment objects are associative maps, in which the key is a string (the name of the environment variable) and the associated data is also a string (the value of the environment variable.)

IEnvironment objects can be iterated, in order to step through all of the environment variables that it holds. It also allows you to add or remove variables, and to modify the contents of existing variables.

Class header file: iprcenv.hpp

IEnvironment - Hierarchy List

IEnvironment

IEnvironment - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IEnvironment objects.

~IEnvironment

```
public:~IEnvironment()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor will destroy the internal list of variable and value strings.

IEnvironment

Constructs an IEnvironment object. The object can optionally be initialized to the existing environment variables from the process' environment.

Overload 1

```
public:IEnvironment(const IEnvironment& toCopy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor replicates the contents of the passed environment object.

toCopy

the source object to copy construct from

Overload 2

```
public:IEnvironment(const bool copyCurrentEnv)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version will create the environment object and optionally initialize it to hold all of the process' current environmental variables. Use this to pass along your own environment to your child processes.

copyCurrentEnv

indicates whether the object should initialize itself to represent the process' current environment

Overload 3

```
public:IEnvironment()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor creates an empty environment object, with no initial variables.

Assignment

Use these member functions to assign one environment object to another.

operator =

```
public:void operator =(const IEnvironment& toAssign)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes any existing contents and duplicates the variables of the source object being assigned.

toAssign
the source object whose contents are to be replicated.

Iterator Methods

Use these member functions to iterate through an IEnvironment object.

first

```
public:bool first(IText& keyText, IText& valueText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the internal iterator to the first variable and returns its name and value.

This is required before calling next() to continue iterating through the variables, or any time that you wish to restart the iteration back to the head of the list.

keyText
filled with the name of the first variable

keyValue
filled with the value of the first variable

next

Moves the internal iterator to the next node and returns its name and value.

```
public:bool next(IText& keyText, IText& valueText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the internal iterator to the next node and returns its name and value.

After calling first() to find the first variable in the environment object, next() may be called to find subsequent variables. When there are no more variables available, next() will return false.

keyText
filled with the name of the next variable

valueText
filled with the value of the next variable

Member Access

Use these member functions to access data members of IEnvironment objects.

add

```
public:void add(const IText& keyText, const IText& valueText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds a new variable to the environment object. The new variable is given the passed value. Values are not necessarily iterated in the order that they are added.

keyText
the name of the variable to add

valueText
the text value to give to this new variable.

clear

```
public:void clear()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Clears out all of the existing variables from the environment object, leaving it as though it was just default constructed.

getCount

```
public:ioc::unumeric_t getCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the count of variables in the environment object.

remove

```
public:bool remove(const IText& keyText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds a variable by its key name (the name of the variable), and removes it from the environment object. The count of variables is decremented to account for this removal. If the name is not found, then the return is false.

If the node removed is the current internal iterator node, then the internal iterator is moved up to the next node so that it remains valid (unless the node removed was the last node in the list or the only node.)

keyText
is the name of the variable to remove.

update

```
public:bool update(const IText& keyText, const IText& newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds a variable by its key name (the name of the variable) and updates its associated text value to the passed new text value. If the variable is not found, then the return is false.

keyText
the name of the variable to update

newValue
the new value text to assign to this variable.

Search Methods

Use these member functions to find particular items in an IEnvironment object.

find

Looks up a variable in the environment object and returns its value, either by an IText return value or by copying the value into a provided IText

object. The latter is more efficient if you already have a target text object, avoiding the need for the creation of a temporary for return.

Overload 1

```
public:IText find(const IText& keyText) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version takes a key name (the name of the variable to find), and returns its contents in a temporary IText object. If it is not found, then the returned object contains an empty string.

keyText
the name of the variable to find

Overload 2

```
public:bool find(const IText& keyText, IText& valueText) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version takes a key (the variable to find) and copies the contents of that variable's value into a passed IText object.

keyText
the name of the variable to find

ValueText
filled with the value of the key, if it is found.

IEnvironment - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEventData

The IEventData class encapsulates the data of an event. IEventData is interchangeable with the following classes, which are actually aliases (that is, typedefs) for IEventData:

- IEventParameter1
- IEventParameter2
- IEventResult
- IHighEventParameter
- ILowEventParameter

AIX Considerations

Although the Open Class Library sometimes internally uses the IEventResult value returned by the application, this value is not returned to the X-Motif window system. X-Motif does not support returning any value. Particularly, it does not support the ability to prevent further processing of the event by the windowing system.

Class header file: ievtdata.hpp

IEventData - Hierarchy List

IEventData

IEventData - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

IEventData

Constructs an IEventData object.

Overload 1

```
public:IEventData(int value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object from an integer. The event data is set to the specified integer. This constructor accepts an uncasted value of 0.

Overload 2

```
public:IEventData(void* value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object from a pointer to a void. The event data is set to the specified void pointer.
value A void pointer to use as the event data.

Overload 3

```
public:IEventData(unsigned short lowValue, unsigned short hiValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object from two unsigned short integer values. The event data is set to an unsigned long integer whose two words are the two specified unsigned short integers.
lowValue
The low 16 bits of the event data
lowValue
The high 16 bits of the event data

Overload 4

```
public:IEventData(BooleanConstants value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object from a boolean value. The event data is set to the specified boolean value.

Overload 5

```
public:IEventData( unsigned short lowValue,
                   char lowByte,
                   char hiByte )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object using the specified unsigned short value and two characters. The event data is set to an unsigned long integer whose two words are the unsigned short integer in the low word and the two characters in the high word.

lowValue
The low 16 bits of the event data

lowByte
The low 8 bits for the high 16 bits of the event data

hiByte The high 8 bits for the high 16 bits of the event data

Overload 6

```
public:IEventData(unsigned long value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object from an unsigned long integer value. The event data is set to the specified unsigned long integer.

value A long value to construct from

Overload 7

```
public:IEventData()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create an IEventData object with the event data set to 0.

Contents

Use these member functions to query and set the event data contained by objects of this class.

char1

```
public:char char1() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the low byte of the event data's low word. This is the same as lowLowByte.

char2

```
public:char char2() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the high byte of the event data's low word. This is the same as lowHighByte.

char3

```
public:char char3() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the low byte of the event data's high word. This is the same as highLowByte.

char4

```
public:char char4() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the high byte of the event data's high word. This is the same as highHighByte.

highHighByte

```
public:char highHighByte() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the high byte of the event data's high word. This is the same as char4.

highLowByte

```
public:char highLowByte() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the low byte of the event data's high word. This is the same as char3.

highNumber

```
public:unsigned short highNumber() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data's high word. This is the same as number2.

lowHighByte

```
public:char lowHighByte() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the high byte of the event data's low word. This is the same as char2.

lowLowByte

```
public:char lowLowByte() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character in the low byte of the event data's low word. This is the same as char1.

lowNumber

```
public:unsigned short lowNumber() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data's low word. This is the same as number1.

number1

```
public:unsigned short number1() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data's low word. This is the same as lowNumber.

number2

```
public:unsigned short number2() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data's high word. This is the same as highNumber.

Conversion

These member functions convert an IEventData object to a four-byte value. You can either explicitly convert the object by calling a member function or casting, or allow the compiler to implicitly convert the object using an operator.

asLong

```
public:long asLong() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the long event data value.

asUnsignedLong

```
public:unsigned long asUnsignedLong() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the unsigned long event data value.

operator char *

```
public:operator char *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data as a pointer to a character.

operator struct HACCEL__ *

```
public:operator struct HACCEL__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HACCEL__ structure, which is equivalent to the HACCEL Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HACCEL.

This operator provides source-level compatibility for code that converts an IEventData object to an IAccelTblHandle object or HACCEL type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HBITMAP__ *

```
public:operator struct HBITMAP__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HBITMAP__ structure, which is equivalent to the HBITMAP Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HBITMAP.

This operator provides source-level compatibility for code that converts an IEventData object to an IBitmapHandle object or HBITMAP type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HDC__ *

```
public:operator struct HDC__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HDC__ structure, which is equivalent to the HDC Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HDC.

This operator provides source-level compatibility for code that converts an IEventData object to an IPresSpaceHandle object or HDC type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HFONT__ *

```
public:operator struct HFONT__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HFONT__ structure, which is equivalent to the HFONT Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HFONT.

This operator provides source-level compatibility for code that converts an IEventData object to an HFONT type, written with an earlier version of the Windows SDK.

operator struct HICON__ *

```
public:operator struct HICON__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HICON__ structure, which is equivalent to the HICON Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HICON.

This operator provides source-level compatibility for code that converts an IEventData object to an IPointerHandle object or HICON type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HINSTANCE__ *

```
public:operator struct HINSTANCE__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HINSTANCE__ structure, which is equivalent to the HINSTANCE Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HINSTANCE.

This operator provides source-level compatibility for code that converts an IEventData object to an IModuleHandle object or HINSTANCE type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HMENU__ *

```
public:operator struct HMENU__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HMENU__ structure, which is equivalent to the HMENU Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HMENU.

This operator provides source-level compatibility for code that converts an IEventData object to an IMenuHandle object or HMENU type, written with an earlier version of the Windows SDK.

operator struct HRGN__ *

```
public:operator struct HRGN__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HRGN__ structure, which is equivalent to the HRGN Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HRGN.

This operator provides source-level compatibility for code that converts an IEventData object to an IRegionHandle object or HRGN type, written with an earlier version of the Windows SDK or ported from another platform.

operator struct HWND__ *

```
public:operator struct HWND__ *() const
```

This is supported on **WIN**

Returns the event data as a pointer an HWND__ structure, which is equivalent to the HWND Windows SDK handle type when STRICT is defined.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND and HACCEL.

This operator provides source-level compatibility for code that converts an IEventData object to an IWindowHandle object or HWND type, written with an earlier version of the Windows SDK or ported from another platform.

operator unsigned long

```
public:operator unsigned long() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the event data as an unsigned long number.

operator void *

```
public:operator void *() const
```

This is supported on **WIN**

Returns the event data as a pointer to non-specific data.

This operator is only available when compiling with the STRICT macro defined. The Windows SDK now defines STRICT by default. Previous versions of the Windows SDK did not define STRICT by default. Compiling with STRICT defined provides greater type safety in the Windows handle types, such as HWND.

This operator provides source-level compatibility for code that converts an IEventData object to an IHandle object (or an object some of some of the other handle classes), written with an earlier version of the Windows SDK or ported from another platform.

IEventData - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IExternalProcess

IExternalProcess is a mechanism for executing and controlling external processes. It provides the functionality traditionally associated with C runtime library functions such as spawn() or exec(), but does so within a fully Open Class Library enabled environment. Each instance of IExternalProcess can control a single process. It allows you to start, terminate, wait for, control the priority of, and retrieve the termination code of the process.

This class works in conjunction with IEnvironment, which is used to build the environment passed to the new process. Note that, upon construction, an IExternalProcess has no environment. If you want to pass any environment to your new process, you must create an IEnvironment object and call setEnvironment() to set up the new environment. The IEnvironment class has an optional constructor that will fill itself in with the current environment of the calling process. Once you have a copy of your own environment, you can add or remove or modify entries in it before setting the environment on the external process object.

Portability Considerations

OS/2 does not allow you to retrieve the priority class of an external process, nor is killing a process necessarily guaranteed to be successful.

OS/2 is not capable of passing Unicode environment variables or command line parameters to the new process. All such text will be converted from their IText format into ASCII text before being passed to the new process.

Class header file: iextproc.hpp

IExternalProcess - Hierarchy List

IExternalProcess

IExternalProcess - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct external process objects.

~IExternalProcess

```
public:~IExternalProcess()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor cleans up any system resources used to refer to the external process and any internal data required to track it. This operation does not imply any attempt to stop the external process. It continues to run until it dies a natural death; you just lose you ability to control it.

IExternalProcess

```
public:IExternalProcess()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the external process object. It does not start the process, and it does not set the path to the executable to be run.

This default constructor is the only available constructor.

Member Access

Use these member functions to access member data of an external process object.

getId

```
public:IProcessId getId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the process id of the external process. If the process is not running, then the results are undefined.

getPath

```
public:IPathName getPath() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the currently set path of this object. The path refers to the path to the executable program that is to be run. It does not have to be set during construction so it may return an empty path if it has never been set.

id

```
public:inline IProcessId id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is an obsoleted method that is supported for backwards compability. Use getId() instead.

setEnvironment

```
public:void setEnvironment(const IEnvironment& newEnv)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the environment of the external process. It cannot be called while the process is running. However, before the process is started or after it terminates, this may be called to set up a new environment for the next time the process is started.

newEnv

the new environment to set. The next time that start() is called, this will be the default evironment to use.

Exception

Exception	Description
IInvalidRequest	if the process is already started.

setPath

```
public:void setPath(const IPathName& newName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the path to the external process. It cannot be called while the process is running. However, before the process is started or after it terminates, this may be called to set up a new path for the next time the process is started.

newName

the new path name to set. The next time that start() is called, this will be the default path to execute.

Exception

Exception	Description
IInvalidRequest	if the process is already started.

Process Control

Use these member functions to control the external process.

kill

```
public:virtual void kill()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method will attempt to kill the process. This is not a recommended action, as it might stop the process such that data or resources are left in an unpredictable state. It also is not necessarily guaranteed to work on all platforms. Some programs may not be killable.

Exception

Exception	Description
IAccessError	if the system API used to implement this method fails.

setPriority

```
public:virtual bool
  setPriority( INonGUIApplication::PriorityClass priority )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the priority class of the process. This only works if the process is actually running. Returns true if the priority was successfully set, else false.

priority

is the new priority class to set for the process.

Exception

Exception	Description
IAccessError	if the system API used to implement this method fails.

start

This method, in all its various variations, starts up the process. This can only be called if the process is not already started. The version with no parameters assumes that all information required has already been set up.

Overload 1

```
public:virtual void start()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assumes that all of the required information is already set, and starts the process without any parameters.

Exception

Exception	Description
IInvalidRequest	if the process is already started.

Exception	Description
IInvalidParameter	if the path currently set is empty or beyond the maximum legal platform length, or does not represent a valid executable file.
IResourceExhausted	if the internal environment buffer used to pass environment information to the new process is overflowed by too much environment variable information.
IAccessError	if the system API used to execute the new process fails.

Overload 2

```
public:virtual void start(const IPathName& path)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the path to the process, and then starts the program without any parameters.

path the path of the process to start. This overrides any existing path that was previously set, even if the start fails.

Exception

Exception	Description
IInvalidRequest	if the process is already started.
IInvalidParameter	if the path currently set is empty or beyond the maximum legal platform length, or does not represent a valid executable file.
IResourceExhausted	if the internal environment buffer used to pass environment information to the new process is overflowed by too much environment variable information.
IAccessError	if the system API used to execute the new process fails.

Overload 3

```
public:virtual void
  start( const IPathName& path,
         const IText* argArray,
         const ioc::unumeric_t argCount )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:virtual void
  start( const IText* argArray,
         const ioc::unumeric_t argCount )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

stop

```
public:inline void stop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is an obsoleted method maintained for backwards compatibility. Use the kill() method instead.

Exception

Exception	Description
IAccessError	if the system API used to implement this method fails.

wait

```
public:virtual bool wait(const ioc::numeric_t timeout = - 1) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Waits for the process to die. It will wait for up to the passed number of milliseconds. The default value, if no timeout is provided, is -1, which means never timeout. After this call is successfully executed, the exitStatus() method can be called to query the exit status of the process. Also, after a successful wait(), it is guaranteed to be safe to call the other methods that are only valid when the process is not running.

Returns true if the process died within the timeout period, else false if the timeout occured.

timeout

the number of milliseconds to wait for the process to die before giving up.

Process Information

Use these member functions to query information about the external process.

exitStatus

```
public:virtual bool exitStatus(ioc::unumeric_t& status) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the exit status of the process, if it has terminated. If it has not terminated, then the return value is false and the passed parameter is not changed.

status is filled with the process exit status if the process is running.

getPriority

```
public:virtual INonGUIApplication::PriorityClass
  getPriority() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the priority class of the external process if it is running. If it is not running or the platform is OS/2, then an exception is generated. This method is not supported on all platforms.

Exception

Exception	Description
IAccessError	if the priority cannot be obtained
IInvalidParameter	if the priority returned from the system is not mappable to a standard Open Class Library process priority class.

isRunning

```
public:virtual bool isRunning() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a boolean value that indicates whether the process is currently running or not. As in any multiprocess system, this is only advisory as the state can change after this call returns but before any action can be taken based on the results.

isStarted

```
public:inline bool isStarted() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is an obsoleted method maintained for backwards compability. Use isRunning() instead.

IExternalProcess - Inherited Member Functions and Data

Inherited Public Functions

IExternalProcess			
Definition	**Page Number**	**Definition**	**Page Number**
`makeImplObj`	43		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IHandle

The IHandle class is a wrapper for a generic handle object. You can create objects of this class, although such objects have limited utility.

The Open Class Library provides other handle classes which represent specific types of handles, such as window handles, bitmap handles, or thread identifiers. You use these specific handle classes when calling Open Class Library functions.

The advantage of providing separate classes for each handle type is that they become distinct enough to prevent a user from passing the wrong handle type to a given function.

Class header file: ibhandle.hpp

IHandle - Hierarchy List

IHandle

IHandle - Member Functions and Data by Group

Constructors & Destructor

Use this member function to create an IHandle object.

IHandle

```
public:IHandle(Value value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

Data Members

handle

```
protected:Value handle
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Diagnostics

Use these member functions to obtain diagnostic information.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of the form nnnn.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

Type Conversions

Use these type conversion operators to convert objects of this class to a different type.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

IHandle - Type Definitions

Value

Variation 1

```
typedef void * Value
```

This is supported on **WIN**

Variation 2

```
typedef unsigned long Value
```

This is supported on **OS/2 AIX OS/400 z/OS**

IHandle - Associated Globals

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IInterest

IInterest is used by an observer to register with a notifier. It specifies what types of notification events an observer is interested in. Typically a notifier publishes interests which an observer can use to register for notifications.

Class header file: iintrst.hpp

IInterest - Hierarchy List

IInterest

IInterest - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct IInterest objects.

~IInterest

```
public:virtual ~IInterest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IInterest object.

IInterest

Constructs an IInterest object.

Overload 1

```
public:IInterest(const INotifier&, const INotificationId&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IInterest object from a notifier and a notification id. The notifier is a reference to the notifier from which the notification originates. The notifier id identifies the type of interest.

Overload 2

```
public:IInterest(const IInterest& src)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy construct an IInterest. This will clear the observerlist cache used by the convenience methods.

Overload 3

```
public:IInterest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an IInterest object for all types of notification events.

operator =

```
public:IInterest& operator =(const IInterest& rhs)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the value of one IInterest object to another.

Comparison Operators

Use these member functions to compare one IInterest object to another.

operator !=

```
public:bool operator !=(const IInterest& theInterest) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another interest object for inequality. Returns true if this object is not equal to the passed object, otherwise returns false.

operator <

```
public:bool operator <(const IInterest& rhs) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another interest object for relative magnitude. The parameter is the interest to compare to this interest. Returns true if this object represents interest that is less than that of the comparison interest.

operator ==

```
public:bool operator ==(const IInterest& theInterest) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another interest object for equality. Returns true if this object is equal to the passed object, otherwise returns false.

Convenience member functions

computeHashValue

```
protected:static ioc::unumeric_t
  computeHashValue( const char* pString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a hashvalue given a string pointer.

Member Access

Use these member functions to access the data of this class.

hashValue

```
public:ioc::unumeric_t hashValue() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the hash value used to compare this interest.

notificationId

```
public:INotificationId notificationId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the notification id associated with this interest.

notifierAddress

```
public:INotifierAddress notifierAddress() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the notifier address associated with this interest.

Observer Notification

These member functions notify observers related to this interest object.

notifyObservers

```
public:virtual void notifyObservers(INotificationEvent&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies any observers registered for this interest.

notifyObserversAsync

```
public:virtual void notifyObserversAsync(INotificationEvent&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Asynchronously notifies any observers registered for this interest.

IInterest - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IModuleHandle

Constructs an IModuleHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

Class header file: ibhandle.hpp

IModuleHandle - Hierarchy List

IModuleHandle

IModuleHandle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct objects of this class.

IModuleHandle

Constructs an IModuleHandle from an operating system dependent value.

Overload 1

```
public:IModuleHandle(Value value = 0)
```

This is supported on **OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IModuleHandle(Value value = 0)
```

This is supported on **WIN**

Windows Considerations

This constructor is available when compiling either without the STRICT macro defined (the STRICT macro is used by the Windows SDK), or with both the STRICT and IC_WIN_STRICT_HANDLES macros defined. In either case, this constructor is intended to accept HINSTANCE and HMODULE values.

If you are compiling without the STRICT macro defined, this constructor will also accept other pointer values. You can provide greater type safety by compiling with both STRICT and IC_WIN_STRICT_HANDLES defined. In this case, this constructor will accept only HINSTANCE and HMODULE values.

Overload 3

```
public:IModuleHandle(Value value = 0)
```

This is supported on **WIN**

Overload 4

```
public:IModuleHandle(void* value = 0)
```

This is supported on **WIN**

Windows Considerations

This constructor is available only when compiling with the STRICT macro defined (the STRICT macro is used by the Windows SDK) and the IC_WIN_STRICT_HANDLES macro not defined. This constructor is intended to accept HINSTANCE or HMODULE values, but may also accept other pointer values. To accept only HINSTANCE and HMODULE values, you must compile your code with both the STRICT and IC_WIN_STRICT_HANDLES macros defined.

Diagnostics

Functions for accessing the handle's value.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

Type Conversions

Use these type conversion operators to convert objects of this class to a different type.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

IModuleHandle - Type Definitions

Value

Variation 1

```
typedef void * Value
```

This is supported on **WIN**

Windows Considerations

This definition makes IModuleHandle::Value equivalent to the HINSTANCE and HMODULE typedefs in the Windows SDK.

This definition is available only when compiling with the STRICT macro defined (the STRICT macro is used by the Windows SDK). Compiling without STRICT defined causes IModuleHandle::Value to be type defined to void*, which allows it to be equivalent to the HINSTANCE and HMODULE typedefs, along with all other handle typedefs in the Windows SDK.

Variation 2

```
typedef unsigned long Value
```

This is supported on **z/OS**

Windows Considerations

This definition makes IModuleHandle::Value equivalent to the HINSTANCE and HMODULE typedefs, as well as all other handle typedefs in the Windows SDK.

This definition is available only when compiling without the STRICT macro defined (the STRICT macro is used by the Windows SDK). Compiling instead with STRICT defined causes IModuleHandle::Value to be type defined to HINSTANCE__*, which makes it equivalent to only the HINSTANCE and HMODULE typedefs in the Windows SDK. This stricter definition provides IModuleHandle with greater type safety.

Variation 3

```
typedef HINSTANCE__ * Value
```

This is supported on **WIN**

Variation 4

```
typedef int Value
```

This is supported on **OS/400**

OS/2 Considerations

This definition makes IModuleHandle::Value equivalent to the HMODULE typedef in the OS/2 Toolkit.

IModuleHandle - Associated Globals

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IModuleHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IModuleHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IModuleHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMRefCounted

IMRefCounted is an abstract base class from which you can derive classes whose objects have reference counts. When an object is constructed, its initial reference count is zero.

When adding and removing references, call the addRef and removeRef functions. Consider using ICountedPointerTo, which is a smart pointer that handles reference counting automatically, if you want addRef and removeRef called implicitly.

An object is deleted when its reference count reaches zero. All IMRefCounted functions are thread safe.

When converting your code to use IMRefCounted, consider that its behavior differs from that of the deprecated IRefCounted class whose reference count is 1-based instead of 0-based.

Class header file: imrefcnt.hpp

IMRefCounted - Hierarchy List

IMRefCounted

IThreadFn

ITimerFn

ITextStyleSet

IStringGeneratorFn

ITextStorage

IBuffer

IMRefCounted - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IMRefCounted objects.

~IMRefCounted

```
protected:virtual ~IMRefCounted()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor deletes the object from the heap. If your class derives from IMRefCounted and it needs to perform cleanup tasks, you should override the destructor. The destructor is called when the reference count is zero.

IMRefCounted

Constructs an IMRefCounted object.

Overload 1

```
protected:IMRefCounted(const IMRefCounted&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a reference-counted object. This function is provided for completeness to give subclassed constructors an inherited method to call. It does not copy the reference count; instead, the reference count of the new object is initialized to zero.

Overload 2

```
protected:IMRefCounted()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a new object with an initial reference count of zero.

operator =

```
protected:IMRefCounted& operator =(const IMRefCounted&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IMRefCounted object to another.

IMRefCounted

A constant reference to the IMRefCounted object to be copied.

Incrementing and Decrementing the Counter

Use these member functions to manage the reference count on the object.

addRef

```
public:void addRef()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds a new reference to the object whose counter is maintained. Use this method when you create a pointer that refers to the object.

removeRef

```
public:void removeRef()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Decrements the reference count to an object. Use this method when you are finished using a pointer to the object. When an object's reference count is zero, the object is deleted.

Obtaining the Current Reference Count

Use these member functions to access information about the reference count of the object.

count

```
public:ioc::unumeric_t count() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of existing references to the object. When the object's reference count is zero, the object is deleted.

IMRefCounted - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

INonGUIApplication

This class represents a non-GUI (Graphical User Interface) application.

Class header file: ingapp.hpp

INonGUIApplication - Hierarchy List

INonGUIApplication

ICurrentNonGUIApplication

INonGUIApplication - Member Functions and Data by Group

Constructors & Destructor

The constructor and destructor for this class are protected. You must derive from this class if you want to use it to represent a non-GUI application.

~INonGUIApplication

```
protected:virtual ~INonGUIApplication()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an INonGUIApplication object. Does not affect the process if it is still running.

INonGUIApplication

```
protected:INonGUIApplication(const IProcessId& id)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an INonGUIApplication object.

Current Information

Use these member functions to query information on the current process.

current

```
public:static ICurrentNonGUIApplication& current()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the calling application.

currentPID

```
public:static IProcessId currentPID()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the process id for the calling process.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a string of formatted debug information for this object.

asString

```
public:virtual IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a string of formatted information on the object.

Member Access

Use these members functions to access data for this class.

id

```
public:virtual IProcessId id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the process id of the process represented by the object.

Priority Management

Use these member functions to manage the priority of the application.

adjustPriority

```
public:virtual INonGUIApplication&
  adjustPriority( long adjustment,
                  bool setDescendents = false )
```

This is supported on **OS/2**

This is a deprecated member function that is only supported on OS/2 for backwards compatibility. It adjusts the priority level.

setPriority

```
public:virtual INonGUIApplication&
  setPriority( PriorityClass priorityClass,
               long priorityLevel = 0,
               bool setDescendents = false )
```

This is supported on **OS/2**

This is a deprecated member function that is only supported on OS/2 for backwards compatibility. It sets the priority level according to the supplied priority class.

setProcessPriority

```
public:virtual INonGUIApplication&
  setProcessPriority( const EProcessPriority newPrio )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Set the priority of the process represented by this object.

INonGUIApplication - Enumerations

EProcessPriority

The process priority levels used by the new, portable priority APIs.

PriorityClass

The priority classes used by the old, OS/2 specific priority APIs.

INonGUIApplication - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

INonGUIApplication			
Definition	**Page Number**	**Definition**	**Page Number**
`setId`	58		

Inherited Protected Data

None

INonGUIThread

The INonGUIThread class represents threads of execution within the current program. You can use this class to implement multithreaded applications. Objects of this class give you access to all of the tasking APIs of the operating system. In addition, these objects can serve as the anchor for thread-specific information.

Generally, you use objects of this class in one of the following ways:

- To apply thread functions to the current thread. In most cases, these functions are applied to the INonGUIThread object reference returned by the static member function INonGUIThread::current
- To create additional threads of execution by creating new objects of this class and starting them.

- To manipulate threads of execution initiated using alternate means, for which only the thread's identifier is known.

Class header file: ingthrd.hpp

INonGUIThread - Hierarchy List

INonGUIThread

ICurrentNonGUIThread

INonGUIThread - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class.

~INonGUIThread

```
public:virtual ~INonGUIThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deallocates thread-related resources. When objects of this class are destructed, the thread is not terminated.

INonGUIThread

Constructs a non-GUI thread object.

This class has four general types of constructors:

- The default constructor, which only creates a C++ object; you must call the thread object's start function to start the thread
- Constructors that accept a thread function and function parameter, which immediately start a new thread
- A constructor that lets you represent an already-running thread
- The copy constructor

Overload 1

```
public:INonGUIThread( OptlinkFnPtr function,
                      void* functionArgument )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of this class when you need to specify a function to run that is defined with the _Optlink calling convention. Such functions are typically started using the function _beginThread.

Use this form of the constructor to create a new object of this class and immediately dispatch it. This is equivalent to using the default constructor and then dispatching the thread using INonGUIThread::start.

function

The _Optlink style function to use as the function pointer for this thread.

functionArgument

The parameter to pass to the thread function.

Exception

Exception	Description
IAccessError	An error occurred trying to start the thread.

POSIX Considerations

This constructor creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

Overload 2

```
public:INonGUIThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Use this constructor to create an object of this class without creating an additional thread of execution. You can subsequently start execution of the thread using INonGUIThread::start.

Overload 3

```
public:INonGUIThread(const INonGUIThread& thread)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor if you would like to make a copy of an existing object of this class. This is commonly referred to as a copy constructor.

Note that copying an INonGUIThread object does not start a new thread of execution. The newly copied INonGUIThread will refer to the same thread of execution as the original (source in the copy) INonGUIThread object.

thread The source thread object used to construct the new thread object.

Overload 4

```
public:INonGUIThread( const ICountedPointerTo < IThreadFn >& function )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create a INonGUIThread object and start a new thread running. The code to be run in the new thread is specified by the run function of the IThreadFn object passed in as a parameter.

The IThreadFn object is reference counted, since IThreadFn is derived from IMRefCounted.

function

Specifies the code to be executed in the new thread.

POSIX Considerations

This constructor creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

Overload 5

```
public:INonGUIThread( SystemFnPtr function,
              ioc::unumeric_t functionArgument )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this form of the constructor to create a new object of this class and immediately dispatch it. This is equivalent to using the default constructor and then dispatching the thread using INonGUIThread::start.

function

The _System style function to use as the function pointer for this thread.

functionArgument

The parameter to pass to the thread function.

POSIX Considerations

This constructor creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

Overload 6

```
public:INonGUIThread( const IThreadId& threadID,
              const IThreadHandle& threadHandle = IThreadHandle ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of this class from the thread identifier (ID) of a previously started thread. You can use this constructor to provide this class' functionality to threads created using alternate means (for example, the native operating system's system calls).

threadID

The thread id of the thread to wrap.

threadHandle

The handle of the thread to wrap.

OS/2 Considerations

The **threadHandle** parameter is ignored.

Windows Considerations

You must specify the **threadHandle** argument if the **threadId** argument does not refer to the current thread. You can use the return value of the CreateThread API for the thread handle.

POSIX Considerations

The **threadHandle** parameter is ignored.

Deprecated Priority Methods

These member functions are deprecated, and represent the old OS/2-centric thread priority scheme. These have been replaced by more portable priority methods.

adjustPriority

```
public:virtual INonGUIThread& adjustPriority(int delta)
```

This is supported on **OS/2**

Changes the thread's priority level by the specified amount. This is a deprecated API that is only supported on OS/2, for backwards compatibility.

delta The amount to adjust the thread priority level by.

Exception

Exception	Description
IAccessError	The thread priority was not adjusted. The priority **delta** may be invalid. The valid values are -31 to 31.

convertedPriority

```
public:static int
  convertedPriority( const int toConvert,
                     INonGUIApplication::PriorityClass& classToSet )
```

This is supported on **OS/2**

Converts a stored priority level into the old style OS/2 specific priority class and level. This is only for use on OS/2, to maintain backwards compatibility.

toConvert

The stored priority to convert.

classToSet

The OS/2 priority class to which the passed priority maps.

priorityClass

```
public:virtual INonGUIApplication::PriorityClass
  priorityClass() const
```

This is supported on **OS/2**

Returns the priority class of this thread. The return value is an enumerator provided by INonGUIApplication::PriorityClass. This is a deprecated method and supported only on OS/2, for backwards compatibility purposes.

priorityLevel

```
public:virtual unsigned priorityLevel() const
```

This is supported on **OS/2**

Returns the priority level of this thread. The return value is between 0 and 31. This is a deprecated method, and is only supported on OS/2, for backwards compatibility reasons.

setPriority

```
public:virtual INonGUIThread&
  setPriority( INonGUIApplication::PriorityClass priority,
               unsigned level )
```

This is supported on **OS/2**

Sets the priority class and level of this thread. This is a deprecated API supported only on OS/2, for backwards compatibility.

priority

The new priority class to set.

level The new priority level to set.

Exception

Exception	Description
IAccessError	The thread priority was not set. The priority class or level may be invalid.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides debugging information about the class object. Use it to return general diagnostic information about the thread.

asString

```
public:virtual IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides debugging information about the class object. Use it to return a string of the form ″INonGUIThread(tid)″, where **tid** represents the thread identifier.

Posix Detaching

This class creates and starts the thread using Posix thread APIs. These members allow you to control whether Open Class Library detaches the Posix thread object when the thread completes.

By default, Open Class Library detaches Posix thread objects that it creates, to free resources associated with the thread, but this could cause calls to Posix thread APIs in your application to fail.

isAutoDetachThread

```
public:virtual bool isAutoDetachThread() const
```

This is supported on **AIX OS/400 z/OS**

This is ignored on **WIN OS/2**

Returns whether Open Class Library will call the Posix thread API pthread_detach when the thread function completes, to free resources associated with the thread. This function has no effect if the thread function has already completed running.

OS/2 Considerations

This function always returns false, since this class does not use Posix thread APIs on this platform.

Windows Considerations

This function always returns false, since this class does not use Posix thread APIs on this platform.

setAutoDetachThread

```
public:virtual void setAutoDetachThread(bool detach = true)
```

This is supported on **AIX OS/400 z/OS**

This is ignored on **WIN OS/2**

Determines whether Open Class Library will call the Posix thread API pthread_detach when the thread function completes, to free resources associated with the thread. This function has no effect if the thread function has already completed running.

detach Specifies whether Open Class Library should free resources by detaching the created Posix thread object when the thread completes. If you specify a value of false, you are expected to call the Posix thread API pthread_join or pthread_detach to free thread resources.

OS/2 Considerations

This function has no effect, because this class does not use Posix thread APIs on this platform. Thread resources are always freed when the thread completes.

Windows Considerations

This function has no effect, because this class does not use Posix thread APIs on this platform. Thread resources are always freed when the thread completes.

Stack Size

Use these member functions to query and set this thread's stack size.

defaultStackSize

```
public:static ioc::unumeric_t defaultStackSize()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the default stack size for threads in bytes. The Open Class Library uses this value by default when creating threads started by objects of this class.

setDefaultStackSize

```
public:static void setDefaultStackSize(ioc::unumeric_t size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the default stack size for threads in bytes. The Open Class Library uses this value by default when creating threads started by objects of this class.

size The new default stack size.

setStackSize

```
public:virtual INonGUIThread& setStackSize(ioc::unumeric_t size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets this thread's stack size in bytes. If you have already started the thread, this value takes effect only when the thread is stopped and restarted.

size The new stack size to set for this thread.

stackSize

```
public:virtual ioc::unumeric_t stackSize() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns this thread's stack size in bytes. If the stack size was set when the thread was running, it might not reflect the size of the currently running thread. Calling this function only takes affect the next time the thread is started.

Windows Considerations

Returns the value of the stack size that was set with the function INonGUIThread::setStackSize. This value represents the stack size used as the initial stack size when the thread is created. For the primary thread, its value is the same as the initial default stack size. This value does not represent the actual committed stack in use by the thread.

You can control the maximum stack available to your program by using the /STACKSIZE linker directive.

Starting and Stopping Threads

Use these member functions to start or stop threads.

resume

```
public:virtual void resume()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resumes a thread's execution after it has been suspended by a call to suspend.

Note: Calling suspend and resume is not a recommended mechanism for thread control. Resources and conditions provide safer services for managing threads.

Exception

Exception	Description
IAccessError	The thread was not suspended or the thread is invalid.

start

Starts a thread, providing the function that will be the thread's starting point in a number of different ways, and providing thread arguments that will be passed along to the thread function.

Overload 1

```
public:virtual void
  start( const ICountedPointerTo < IThreadFn >& function )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version to start a thread that executes a function represented by an IThreadFn object. The IThreadFn object is reference counted, since IThreadFn is derived from IMRefCounted.

function

The object representing the function to run on the new thread.

functionArgument

The parameter to pass to the thread function.

POSIX Considerations

This function creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

Overload 2

```
public:virtual void
  start( SystemFnPtr function,
         ioc::unumeric_t functionArgument )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of this class when you need to specify a function to run that you have defined using the _System calling convention. Such functions are typically started using the thread APIs that are defined for the operating system.

function

The _System style function to run on the new thread.

functionArgument

The parameter to pass to the thread function.

POSIX Considerations

This function creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

Overload 3

```
public:virtual void
  start( OptlinkFnPtr function,
         void* functionArgument )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version to start a thread that executes a function defined with the _Optlink calling convention. Such functions are typically started using the function _beginThread. This function gives you the same capability.

function

The _Optlink style function to run on the new thread.

functionArgument

The parameter to pass to the thread function.

Exception

Exception	Description
IAccessError	An error occurred trying to start the thread.
IInvalidRequest	The thread has already been started.

POSIX Considerations

This function creates and starts the thread using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t value of the thread.

By default, Open Class Library will detach the Posix thread object, to free resources associated with the thread, when the thread completes. See INonGUIThread::setAutoDetachThread for more details.

stop

```
public:virtual void stop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the thread. This function causes the thread to stop immediately, bypassing some normal termination functions. Use it with extreme caution, preferably when the state of the target thread is known.

Exception

Exception	Description
IAccessError	The thread was not stopped. The thread was either busy or the thread is invalid.

OS/2 Considerations

Do not use this function on suspended threads or the results can be unpredictable.

suspend

```
public:virtual void suspend()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Suspends the thread's execution, which can be undone later with a call to resume.

Note: Calling suspend and resume is not a recommended mechanism for thread control. Resources and conditions provide safer services for managing threads.

Exception

Exception	Description
IAccessError	(OS/2 and Windows) The thread is invalid. The thread was not suspended.
IInvalidRequest	(AIX, OS/390, AS/400) The suspend function is not supported.

POSIX Considerations

This function is not supported. If you call it Open Class Library throws an exception.

newStartedThread

```
protected:static IStartedThread* newStartedThread()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an object of the IStartedThread class.

This function is for the internal use of the Open Class Library. It should not be used by your own derived classes.

Thread Information

Use these member functions to query and set general thread information.

current

```
public:static ICurrentNonGUIThread& current()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an object of the ICurrentNonGUIThread class that represents the currently executing thread.

currentHandle

```
public:static IThreadHandle currentHandle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the thread handle for the current thread.

OS/2 Considerations

The thread handle and thread ID are identical.

Windows Considerations

You must use the thread handle for the operating system thread and synchronization functions.

POSIX Considerations

The thread handle and thread ID are identical.

currentId

```
public:static IThreadId currentId()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the identifier of the currently executing thread.

exitCode

```
public:ioc::unumeric_t exitCode() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the exit code for the thread. If the thread is still running, the value is undefined.

handle

```
public:virtual IThreadHandle handle() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the thread handle for this object's thread.

OS/2 Considerations

The thread handle and thread ID are identical.

Windows Considerations

You must use the thread handle for the operating system thread and synchronization functions.

POSIX Considerations

The thread handle and thread ID are identical.

id

```
public:virtual IThreadId id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the identifier of the thread. A return value of 0 indicates that the thread is not started.

isStarted

```
public:virtual bool isStarted() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines if this thread is currently active. This function returns true if the thread is running, false otherwise.

setVariable

```
public:INonGUIThread&
  setVariable( const IString& key,
               const IString& value )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stores data and an associated key, on a per thread instance. You can later retrieve this data using INonGUIThread::variable.

key The name of the new thread variable to set.

value The new value to give to the new thread variable.

variable

```
public:IString variable(const IString& key) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the data associated with the key that was stored using INonGUIThread::setVariable. If there is no data associated with the key, this function returns a zero-length IString object.

key The name of the thread variable to query.

Thread Priority

Use these member functions to control the thread's priority. You can query or set the priority of a thread.

setThreadPriority

```
public:virtual INonGUIThread&
  setThreadPriority( const INonGUIThread::EThreadPriority newPriority )
```

This is supported on **WIN OS/2 AIX OS/400**

This is ignored on **z/OS**

Sets this thread's priority level.

This is the portable mechanism for setting thread priorities, and replaces the function setPriority.

newPriority

The new portable thread priority level to set for this thread.

Exception

Exception	Description
IAccessError	Failed to set the thread priority.

threadPriority

```
public:virtual INonGUIThread::EThreadPriority
  threadPriority() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a thread's priority level.

This is the portable mechanism for retrieving a thread's priority, and replaces the function priorityClass.

INonGUIThread - Enumerations

EThreadPriority

These values represent the portable thread priority levels available to set via setThreadPriority.

INonGUIThread - Type Definitions

OptlinkFnPtr

```
typedef void ( _Optlink * OptlinkFnPtr)( void * )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This typedef defines a pointer to a function that uses the _Optlink calling convention.

Note: _Optlink is the default calling convention for this compiler.

SystemFnPtr

```
typedef void ( _System * SystemFnPtr)( ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This typedef defines a pointer to a function that uses the _System calling convention.

INonGUIThread - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

INonGUIThread			
Definition	**Page Number**	**Definition**	**Page Number**
`operator =`	61	`startedThread`	61

Inherited Protected Data

None

INonGUIThread::Cursor

The INonGUIThread::Cursor class creates and manages the cursor for an INonGUIThread object. This nested class defines objects that you can use to traverse or iterate through a set of active threads. In the same way that you can use a cursor to traverse the objects in a collection, you can use this cursor to traverse a set of threads one at a time.

Typically, you traverse a set, or collection, of active threads by doing the following:
1. Calling setToFirst
2. Looping through the threads using setToNext
3. Processing the returned thread IDs from threadId until isValid returns false

Note: This class only provides access to threads that are or have been represented by an INonGUIThread object or an object of a derived class, such as IThread.

Class header file: ingthrd.hpp

INonGUIThread::Cursor - Hierarchy List

INonGUIThread::Cursor

INonGUIThread::Cursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this nested class.

~Cursor

```
public:virtual ~Cursor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the cursor object.

Cursor

```
public:Cursor(bool allThreads = true)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs objects of this class.

allThread

This is ignored and is present only for backwards compatibility.

Thread Iteration

Use these member functions to iterate through the set of active threads.

invalidate

```
public:virtual void invalidate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Marks the cursor as invalid.

isValid

```
public:virtual bool isValid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines if the cursor is valid. This function returns true if the cursor is valid. Otherwise it returns false.

setToFirst

```
public:virtual bool setToFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor's position to the first thread. This function returns true if the cursor is valid. Otherwise it returns false.

setToNext

```
public:virtual bool setToNext()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the cursor's position to the next thread. This function returns true if cursor is still valid. Otherwise it returns false.

threadId

```
public:IThreadId threadId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the identifier (ID) of the thread at the current cursor position.
Exception

Exception	Description
IInvalidParameter	The thread identifier (ID) was not returned. The cursor is not valid.

INonGUIThread::Cursor - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

INotificationEvent

The INotificationEvent class provides the details of a notification event to an observer object. INotifier objects create notification events when these objects change or when they must notify observer objects of events.

Class header file: inotifev.hpp

INotificationEvent - Hierarchy List

INotificationEvent

INotificationEventFor

INotificationEvent - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct, and assign objects of this class.

~INotificationEvent

```
public:virtual ~INotificationEvent()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an INotificationEvent object.

INotificationEvent

Constructs an INotificationEvent object.

Overload 1

```
public:INotificationEvent( INotifier& notifier,
                           const INotificationId& identifier )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an INotificationEvent object using a notification identifier and a reference to a notifier object derived from INotifier.

Overload 2

```
public:INotificationEvent(const IInterest& interest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a notification event from an interest. Notifiers typically publish their supported interests through method accessors.

Overload 3

```
public:INotificationEvent(const INotificationEvent& event)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an INotificationEvent object using a copy of an existing notification event.

Overload 4

```
protected:INotificationEvent()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

operator =

```
public:INotificationEvent&
  operator =( const INotificationEvent& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the contents of one INotificationEvent object with another INotificationEvent object.

Duplication

Use this member function to clone INotificationEvent objects.

clone

```
public:virtual INotificationEvent* clone() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to a copy of this object.

Event Attributes

Use these member functions to get and set the attributes of objects of this class.

interest

```
public:virtual const IInterest& interest() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the IInterest which contains event information such as the notifier address and event id.

notificationId

```
public:virtual INotificationId notificationId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the INotificationId for the event. The derived INotifier classes document the notification identifiers.

notifierAddress

```
public:virtual INotifierAddress notifierAddress() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an INotifierAddress used to identify the originator of the event.

INotificationEvent - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

INotificationEventFor

An INotificationEventFor is a specialized subclass of INotificationEvent which allows a notifier to pass event-specific data to its observers.

Class header file: inotifev.hpp

INotificationEventFor - Hierarchy List

INotificationEvent

INotificationEventFor

INotificationEventFor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct, and assign objects of this class.

~INotificationEventFor

```
public:virtual ~INotificationEventFor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and INotificationEventFor object.

INotificationEventFor

Constructs an INotificationEventFor object.

Overload 1

```
public:INotificationEventFor( const INotificationEventFor <
                              AType >& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an INotificationEventFor object from another INotificationEventFor object.

Overload 2

```
public:INotificationEventFor( const IInterest& interest,
                        const AType& data )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an INotificationEvent from an interest and a reference to some event data. The event data is copied within the constructor.

operator =

```
public:INotificationEventFor < AType >&
  operator =( const INotificationEventFor < AType >& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the contents of one INotificationEventFor object to another.

Duplication

Use this member function to clone INotificationEvent objects.

clone

```
public:virtual INotificationEvent* clone() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to a copy of this object.

Event Attributes

Use these member functions to get and set the attributes of objects of this class.

eventData

```
public:const AType& eventData() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the templatized event data associated with the event.

interest

```
public:virtual const IInterest& interest() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the IInterest which contains event information such as the notifier address and event id.

notificationId

```
public:virtual INotificationId notificationId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the INotificationId for the event. The derived INotifier classes document the notification identifiers.

notifierAddress

```
public:virtual INotifierAddress notifierAddress() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an INotifierAddress used to identify the originator of the event.

INotificationEventFor - Inherited Member Functions and Data

Inherited Public Functions

INotificationEvent			
Definition	**Page Number**	**Definition**	**Page Number**
~INotificationEvent	75	INotificationEvent	75
operator =	75		

Inherited Public Data

None

Inherited Protected Functions

INotificationEvent			
Definition	**Page Number**	**Definition**	**Page Number**
INotificationEvent	75		

Inherited Protected Data

None

INotifier

The INotifier class defines the notification protocol for objects that support observation. Because this class is an abstract base class, you cannot construct objects of this class. It is the responsibility of the derived class to do the actual notifications.

You can implement a notification protocol in the following ways:

- Derive a class from the IStandardNotifier class, which inherits from INotifier, for a direct implementation of the INotifier protocol.
- Derive from the INotifier class and implement your own notification protocol.

INotifier objects publish IInterest or INotificationIds for each notification that the derived class provides. You should document the details of these notifications, including if any notifier data is sent with the notification event.

INotifier objects notify their observers of all events after the observer requests notification by calling IObserver::handleNotificationsFor.

Class header file: inotify.hpp

INotifier - Hierarchy List

INotifier

IStandardNotifier

INotifier - Member Functions and Data by Group

Constructors & Destructor

You cannot construct objects of this class because it is an abstract base class.

~INotifier

```
public:virtual ~INotifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an INotifier object.

INotifier

```
public:INotifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an INotifier object.

Notification Identifiers

The following INotificationIds are provided by INotifier objects.

allChangesId

```
public:static INotificationId const allChangesId
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notification identifier provided to observers when an object derived from INotifier is changed. It is the responsibility of the object derived from INotifier to send this notification. The two derived classes which currently send this notification are IWindow and IStandardNotifier.

IInterest objects created using the default constructor register for this type of notification.

deleteId

```
public:static INotificationId const deleteId
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notification identifier provided to observers when an object derived from INotifier is deleted. It is the responsibility of the object derived from INotifier to send this notification. The two derived classes which currently send this notification are IWindow and IStandardNotifier.

IStandardNotifier sends this notification from its destructor. This means that the derived portions of the notifier have already been deleted. You should not, therefore, cast the pointer to the notifier data to an object that is derived from IStandardNotifier.

Notification Members

Use these member functions to affect the ability of the INotifier object to notify observers of events.

disableNotification

```
public:virtual INotifier& disableNotification() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the notifier from sending notifications to its observers.

enableNotification

```
public:virtual INotifier&
  enableNotification( bool enable = true ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts the notifier sending notifications to its observers. This function can be overridden by derived classes to perform customized notification that your application might need. For instance, one of your functions may require that a database be accessible before processing a retrieve function.

isEnabledForNotification

```
public:virtual bool isEnabledForNotification() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if a notifier can send notifications to its observers.

notifierAddress

```
public:virtual INotifierAddress notifierAddress() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a notifier address used to identify this notifier by observers. INotifier provides the default implementation.

Observer Addition and Removal

IObserver objects use these member functions to add and remove themselves from the notifier's collection.

addObserver

```
protected:virtual INotifier&
  addObserver( IObserver& observer,
               const IInterest& interest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function does nothing by default. It is provided for backward compatibility.

removeAllObservers

```
protected:virtual INotifier& removeAllObservers() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all observers from the notifier's list of observers.

removeObserver

Removes an observer from the notifier's list of observers.

Overload 1

```
protected:virtual INotifier&
  removeObserver( IObserver& observer,
                  const IInterest& interest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function does nothing by default. It is provided for backward compatibility.

Overload 2

```
protected:virtual INotifier& removeObserver(IObserver& observer) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes an observer from the notifier's list of observers.

observer
Reference to the IObserver to be removed from the list of observers.

Observer Notification

These member functions notify observers of a change in a notifier.

notifyObservers

Notifies all observers interested in a particular notification event. Each observer receives a notification event containing the identity of the notifier, the notification address.

Note: A public and a protected version of notifyObservers are provided for convenience. The protected version does not require the caller to construct an INotificationEvent to call it. In this case, the construction of the INotificationEvent occurs in the code of the protected notifyObservers function.

Overload 1

```
public:virtual INotifier&
  notifyObservers( const INotificationEvent& event ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers interested in a particular notification event.

event Reference to the INotificationEvent to be sent to the observers.

Overload 2

```
protected:virtual INotifier&
  notifyObservers( const INotificationId& id ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers interested in a particular notification event.

id Reference to the notification identifier to be sent to the observers in the form of an INotificationEvent.

notifyObserversAsync

```
public:virtual INotifier&
  notifyObserversAsync( const INotificationEvent& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function does nothing by default. It is provided for backward compatibility.

Obsolete

These member functions will soon be obsoleted.

observerList

```
protected:virtual IObserverList&
  observerList( const IInterest* anInterest = 0 ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the list of observers. If the observer list does not exist, the derived notifier class must create it before calling this member function.

Thread Information

Use this member function to object information about the thread in which this notifier was created.

threadId

```
public:const IThreadId& threadId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the thread identifier in which this notifier was created.

INotifier - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

INotifierAddress

INotifierAddress is a wrapper class to encapsulate a notifier. INotifierAddress should be used as an address-independent way to reference a notifier.

Clients should not subclass from this class.

Class header file: inotadr.hpp

INotifierAddress - Hierarchy List

INotifierAddress

INotifierAddress - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct and assign INotifierAddress objects.

~INotifierAddress

```
public:virtual ~INotifierAddress()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an INotifierAddress object.

INotifierAddress

Creates an INotifierAddress object and initializes the member data.

Overload 1

```
public:INotifierAddress(const INotifier* notifier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an INotifierAddress object based on an existing INotifier.

Overload 2

```
public:INotifierAddress()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

operator =

```
public:INotifierAddress& operator =(const INotifierAddress& src)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one INotifierAddress object to another.

Comparison and validation

Use these operators to perform equality and inequality comparisons.

operator !=

```
public:bool operator !=(const INotifierAddress& src) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another INotifierAddress object for inequality. Returns true if this object is not equal to the passed object, otherwise returns false.

operator ==

```
public:bool operator ==(const INotifierAddress& src) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another INotifierAddress object for equality. Returns true if this object is equal to the passed object, otherwise returns false.

Member Access

Use these member functions to access data members of INotifierAddress objects.

notifier

```
public:INotifier& notifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the reference to the notifier wrapped by this object.

INotifierAddress - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IObserver

The IObserver class is the abstract base class for all objects that are to be notified of changes in the state of other objects in the system. Connection objects are derived from this class to connect notifiers with a designated target object. Typically you use an IObserverConnectionTo or an IObserverForConnectionTo to wire your class for notifications.

You can create your own connection objects by deriving from this class and implementing the function dispatchNotificationEvent to process specific events.

Class header file: iobservr.hpp

IObserver - Hierarchy List

IObserver

IObserverForConnectionTo

IObserverConnectionTo

IObserver - Member Functions and Data by Group

Constructors & Destructor

Only derived classes can create objects of this class. To enforce this, the constructor has protected access.

~IObserver

```
public:virtual ~IObserver()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IObserver object.

IObserver

```
protected:IObserver()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Event Dispatching

Use these member functions to evaluate events and determine if it is appropriate for an observer object to process them. These members also attach the observer to and detach the observer from the INotifier object.

handleNotificationsFor

Attaches the observer to the INotifier object. The observer is notified of all events after the notifier object has been enabled for notifications.

Overload 1

```
public:virtual IObserver& handleNotificationsFor(IInterest&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The interest is typically published by the notifier.

Overload 2

```
public:virtual IObserver&
  handleNotificationsFor( INotifier&,
                          const INotificationId& id )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:virtual IObserver&
  handleNotificationsFor( INotifier& notifier,
                          const IEventData& userData = IEventData ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

stopHandlingNotificationsFor

Detaches the observer from the INotifier object. If the observer is handling notifications, this method should be called before the observer is destructed.

Overload 1

```
public:virtual IObserver&
  stopHandlingNotificationsFor( const IInterest& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IObserver&
  stopHandlingNotificationsFor( INotifier&,
                                const INotificationId& id )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:virtual IObserver&
  stopHandlingNotificationsFor( INotifier& notifier )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

dispatchNotificationEvent

```
protected:virtual IObserver&
  dispatchNotificationEvent( const INotificationEvent& event ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A notifier calls this dispatch method after the observer has registered to receive events.

Member Access

Use these member functions to access data of this class.

threadId

```
public:const IThreadId& threadId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the thread identifier in which this notifier was created.

Obsolete

These member functions will soon be obsoleted.

handleNotificationsFor

```
public:virtual IObserver&
  handleNotificationsFor( INotifier& notifier,
                          const IEventData& userData = IEventData ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attaches the observer to the INotifier object. The observer is notified of all events after the notifier object has been enabled for notifications.

IObserver - Inherited Member Functions and Data

Inherited Public Functions

IObserver			
Definition	**Page Number**	**Definition**	**Page Number**
`requestProcessor`	84		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IObserverConnectionTo

IObserverConnectionTo is used to wire notifications from a notifier to an observer member function.

Class header file: iobservr.hpp

IObserverConnectionTo - Hierarchy List

IObserver

IObserverConnectionTo

IObserverConnectionTo - Member Functions and Data by Group

Constructors & Destructor

Use these members to construct and destruct objects of this class. You cannot assign or copy objects of this class.

~IObserverConnectionTo

```
public:virtual ~IObserverConnectionTo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys this object.

IObserverConnectionTo

```
public:IObserverConnectionTo( ATarget& target,
                      MemberFunction memberFunction )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IObserverConnectionTo object.

target Target object to receive notifications.

memberFunction

Member function to receive notifications.

The member function is passed an INotificationEvent object which contains the event data.

Event Dispatching

Use these member functions to evaluate events and determine if it is appropriate for an observer object to process them.

dispatchNotificationEvent

```
protected:virtual IObserverConnectionTo < ATarget >&
  dispatchNotificationEvent( const INotificationEvent& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Dispatches the notification event to the target member function.

IObserverConnectionTo - Type Definitions

MemberFunction

```
typedef void ( ATarget::* MemberFunction)( const INotificationEvent & )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IObserverConnectionTo - Inherited Member Functions and Data

Inherited Public Functions

IObserver			
Definition	**Page Number**	**Definition**	**Page Number**
~IObserver	84	handleNotificationsFor	85
requestProcessor	84	stopHandlingNotificationsFor	85
threadId	86		

Inherited Public Data

None

Inherited Protected Functions

IObserver			
Definition	**Page Number**	**Definition**	**Page Number**
`dispatchNotificationEvent`	85	`IObserver`	84

Inherited Protected Data

None

IObserverForConnectionTo

IObserverForConnectionTo is used to wire notifications with event data from a notifier to a target member function. This connection is activated by registering with an IInterest reference, typically published by a notifier.

Class header file: iobservr.hpp

IObserverForConnectionTo - Hierarchy List

IObserver

IObserverForConnectionTo

IObserverForConnectionTo - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class. You cannot assign or copy objects of this class.

~IObserverForConnectionTo

```
public:virtual ~IObserverForConnectionTo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys this object.

IObserverForConnectionTo

```
public:IObserverForConnectionTo( ATarget& target,
                                 MemberFunction memberFunction )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Contructs an IObserverForConnectionTo object.

target Target object to receive notifications.

memberFunction

Member function to receive notifications.

The member function is passed an INotificationEventFor object which contains the event data.

Event Dispatching

Use these member functions to evaluate events and determine if it is appropriate for an observer object to process them.

dispatchNotificationEvent

```
protected:virtual IObserverForConnectionTo < AType , ATarget >&
  dispatchNotificationEvent( const INotificationEvent& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Dispatches the notification event to the target member function.

IObserverForConnectionTo - Type Definitions

MemberFunction

```
typedef void ( ATarget::* MemberFunction)( const INotificationEventFor <
                                           AType > & event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IObserverForConnectionTo - Inherited Member Functions and Data

Inherited Public Functions

IObserver			
Definition	**Page Number**	**Definition**	**Page Number**
~IObserver	84	handleNotificationsFor	85
requestProcessor	84	stopHandlingNotificationsFor	85
threadId	86		

Inherited Public Data

None

Inherited Protected Functions

IObserver			
Definition	**Page Number**	**Definition**	**Page Number**
dispatchNotificationEvent	85	IObserver	84

Inherited Protected Data

None

IObserverList

The IObserverList class provides the interface for a list of IObserver objects. This class implements the list of observers as an ordered list.

Class header file: iobslist.hpp

IObserverList - Hierarchy List

IObserverList

IObserverList - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct objects of this class.

~IObserverList

```
public:virtual ~IObserverList()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys this object.

IObserverList

Construct an IObserverList object.

Overload 1

```
public:IObserverList(const IObserverList& obsList)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The IObserverList object is created based on the contents of another IObserverList object.

Overload 2

```
public:IObserverList()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor that takes no arguments.

Assignment

Assignment operators.

operator =

```
public:IObserverList& operator =(const IObserverList& obsList)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the contents of one IObserverList object to another.

Observer Addition and Removal

Use these member functions to add, remove, and find IObserver objects in the observer list's collection.

add

```
public:virtual bool add(IObserver& observer, void* userData)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds an observer to the end of the list.

isEmpty

```
public:bool isEmpty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if there are no observers in the list.

remove

```
public:virtual IObserverList& remove(const IObserver& observer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the specified observer from the list.

removeAll

```
public:virtual IObserverList& removeAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all observers from the list.

Observer Notification

These member functions notify observers of a change in a notifier.

notifyObservers

```
public:IObserverList&
  notifyObservers( const INotificationEvent& event )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Traverses the list of observers and calls each member's dispatchNotificationEvent function passing a specified notification event object.

notifyObserversAsync

```
public:IObserverList&
  notifyObserversAsync( const INotificationEvent& anEvent )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Traverses the list of observers and posts a notification request onto the observer's thread which calls each member's dispatchNotificationEvent function passing a specified notification event object.

IObserverList - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

ioc

This namespace contains typedef's which are used throughout the IOC library.

Class header file: ilanglvl.hpp

ioc - Hierarchy List

ioc

ioc - Member Functions and Data by Group

ioc - Enumerations

EIocMsgConst

IOC message string constants. For IOC library use only.

EWIocMsgConst

IOC wide character string constants. For IOC library use only.

uMVSValues

ASCII values used by IOC library.

ioc - Type Definitions

idirent_lg_t

```
typedef struct dirent_lg idirent_lg_t
```

This is supported on **OS/400**

IOC library equivalent of C typedef dirent_lg .

idirent_t

```
typedef struct dirent idirent_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct dirent .

iitimerval_t

```
typedef struct iitimerval { struct itimeval it_interval ;
  struct itimeval it_value ; } iitimerval_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct itimerval .

ioffset_t

IOC library equivalent of C typedef offset_t .

Variation 1

```
typedef long long ioffset_t
```

This is supported on **OS/400**

Variation 2

```
typedef offset_t ioffset_t
```

This is supported on **WIN OS/2 AIX**

iptrdiff_t

IOC library equivalent of C typedef ptrdiff_t .

Variation 1

```
typedef signed long long iptrdiff_t
```

This is supported on **OS/400**

Variation 2

```
typedef ptrdiff_t iptrdiff_t
```

This is supported on **WIN OS/2 AIX z/OS**

isembuf_t

```
typedef struct sembuf isembuf_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct sembuf .

isigaction_t

```
typedef struct sigaction isigaction_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct sigaction .

isigset_t

```
typedef sigset_t isigset_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C typedef sigset_t .

isize_t

IOC library equivalent of C typedef size_t .

Variation 1

```
typedef size_t isize_t
```

This is supported on **WIN OS/2 AIX z/OS**

Variation 2

```
typedef unsigned long long isize_t
```

This is supported on **OS/400**

issize_t

IOC library equivalent of C typedef ssize_t .

Variation 1

```
typedef signed long long issize_t
```

This is supported on **OS/400**

Variation 2

```
typedef ssize_t issize_t
```

This is supported on **WIN OS/2 AIX z/OS**

istat_t

IOC library equivalent of C struct stat .

Variation 1

```
typedef struct stat istat_t
```

This is supported on **WIN OS/2 AIX z/OS**

Variation 2

```
typedef struct istat { mode_t st_mode ; ino_t st_ino ;
        uid_t st_uid ; gid_t st_gid ;
        off_t st_size ; itime_t st_atime ; itime_t st_mtime ;
        itime_t st_ctime ; dev_t st_dev ; isize_t st_blksize ;
        nlink_t st_nlink ; unsigned short st_codepage ;
        unsigned long long st_allocsize ; unsigned int st_ino_gen_id ;
        qp0l_objtype_t st_objtype ; char st_reserved1 [ 55 ] ;
        unsigned short st_ccsid ; } istat_t
```

This is supported on **OS/400**

istat64_t

IOC library equivalent of C struct stat64 .
Variation 1

```
typedef struct stat64 istat64_t
```

This is supported on **OS/400**
Variation 2

```
typedef struct stat istat64_t
```

This is supported on **WIN OS/2 AIX z/OS**

istatfs_t

```
typedef struct statfs istatfs_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct statfs .

istatvfs_t

```
typedef struct statvfs istatvfs_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct statvfs .

istatvfs64_t

```
typedef struct statvfs64 istatvfs64_t
```

This is supported on **OS/400**

IOC library equivalent of C struct statvfs64 .

itime_t

IOC library equivalent of C typedef time_t .
Variation 1

```
typedef signed long long itime_t
```

This is supported on **OS/400**
Variation 2

```
typedef time_t itime_t
```

This is supported on **WIN OS/2 AIX z/OS**

itm_t

```
typedef struct tm itm_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct tm .

iutimbuf_t

```
typedef struct utimbuf iutimbuf_t
```

This is supported on **WIN OS/2 AIX z/OS**

IOC library equivalent of C struct utimbuf .

new_IMSGTEXT_RTVM0100_t

```
typedef _Packed struct { ioc::numeric_t Bytes_Return ;
                         ioc::numeric_t Bytes_Available ;
                         ioc::numeric_t Length_Message_Returned ;
```

```
                        ioc::numeric_t Length_Message_Available ;
                        ioc::numeric_t Length_Help_Returned ;
                        ioc::numeric_t Length_Help_Available ;
                        char Message [ 1 ] ; }
        new_IMSGTEXT_RTVM0100_t
```

This is supported on **OS/400**

Typedef used when calling IOC API QMHRTVM.

numeric_t

Describes a 64-bit signed integer data type in a platform independent way.

Variation 1

```
typedef signed long long numeric_t
```

This is supported on **OS/400**

Variation 2

```
typedef signed long numeric_t
```

This is supported on **WIN OS/2 AIX z/OS**

old_IMSGTEXT_RTVM0100_t

```
typedef _Packed struct { int Bytes_Return ; int Bytes_Available ;
int Length_Message_Returned ; int Length_Message_Available ;
int Length_Help_Returned ; int Length_Help_Available ;
char Message [ 1 ] ; } old_IMSGTEXT_RTVM0100_t
```

This is supported on **OS/400**

Typedef used when calling system API QMHRTVM.

pnstruct_t

```
typedef struct pnstruct { Qlg_Path_Name_T qlg_struct ;
  unichar_t pathname [ 256 ] ; } pnstruct_t
```

This is supported on **OS/400**

IOC library equivalent of C typedef Qlg_Path_Name_T along with space for the path name .

UCS2Char

```
typedef unsigned short UCS2Char
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Describes a Unicode character type.

UniChar

Describes a Unicode character type whose length varies depending on the operating system.

Variation 1

```
typedef unsigned int UniChar
```

This is supported on **AIX**

Variation 2

```
typedef unsigned short UniChar
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

unichar_t

Describes a Unicode character type whose length varies depending on the operating system.

Variation 1

```
typedef wchar_t unichar_t
```

This is supported on **WIN OS/2 OS/400 z/OS**

Variation 2

```
typedef unsigned short unichar_t
```

This is supported on **WIN OS/2 OS/400 z/OS**

unumeric_t

Describes a 64-bit unsigned integer data type in a platform independent way.

Variation 1

```
typedef unsigned long long unumeric_t
```

This is supported on **OS/400**

Variation 2

```
typedef unsigned long unumeric_t
```

This is supported on **WIN OS/2 AIX z/OS**

ioc - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

ioc::itimeval

IOC library equivalent of C struct timeval .

Class header file: ilanglvl.hpp

ioc::itimeval - Hierarchy List

ioc::itimeval

ioc::itimeval - Member Functions and Data by Group

Miscellaneous Members

tv_sec

```
public:long long tv_sec
```

This is supported on **WIN OS/2 AIX z/OS**

tv_usec

```
public:long long tv_usec
```

This is supported on **WIN OS/2 AIX z/OS**

ioc::itimeval - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

ioc::itimezone

IOC library equivalent of C struct timezone .

Class header file: ilanglvl.hpp

ioc::itimezone - Hierarchy List

ioc::itimezone

ioc::itimezone - Member Functions and Data by Group

Miscellaneous Members

tz_dsttime

```
public:long long tz_dsttime
```

This is supported on **WIN OS/2 AIX z/OS**

tz_minuteswest

```
public:long long tz_minuteswest
```

This is supported on **WIN OS/2 AIX z/OS**

ioc::itimezone - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

ioc::statvfs

IOC library struct used as one of the parameters to C function statvfs .

Class header file: ilanglvl.hpp

ioc::statvfs - Hierarchy List

ioc::statvfs

ioc::statvfs - Member Functions and Data by Group

Miscellaneous Members

f_basetype

```
public:char f_basetype [ 80 ]
```

This is supported on **OS/400**

f_bavail

```
public:isize_t f_bavail
```

This is supported on **OS/400**

f_bfree

```
public:isize_t f_bfree
```

This is supported on **OS/400**

f_blocks

```
public:isize_t f_blocks
```

This is supported on **OS/400**

f_bsize

```
public:isize_t f_bsize
```

This is supported on **OS/400**

f_dirlinkmax

```
public:issize_t f_dirlinkmax
```

This is supported on **OS/400**

f_favail

```
public:isize_t f_favail
```

This is supported on **OS/400**

f_ffree

```
public:isize_t f_ffree
```

This is supported on **OS/400**

f_files

```
public:isize_t f_files
```

This is supported on **OS/400**

f_flag

```
public:isize_t f_flag
```

This is supported on **OS/400**

f_frsize

```
public:isize_t f_frsize
```

This is supported on **OS/400**

f_fsid

```
public:isize_t f_fsid
```

This is supported on **OS/400**

f_namemax

```
public:isize_t f_namemax
```

This is supported on **OS/400**

f_objlinkmax

```
public:issize_t f_objlinkmax
```

This is supported on **OS/400**

f_pathmax

```
public:isize_t f_pathmax
```

This is supported on **OS/400**

f_reserved1

```
public:char f_reserved1 [ 12 ]
```

This is supported on **OS/400**

ioc::statvfs - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPerformanceSettings

The IPerformanceSettings class defines settings which you can change on an application wide basis to affect your application's performance. The enumeration IPerformanceSettings::Setting describes the settings you can change using this class.

Class header file: iperfset.hpp

IPerformanceSettings - Hierarchy List

IPerformanceSettings

IPerformanceSettings - Member Functions and Data by Group

Settings

Use these member functions to tune Open Class Library to your application's requirements for thread safety and error checking.

disableSetting

```
public:static void disableSetting(Setting setting)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to disable the IPerformanceSettings::Setting specified by setting.

enableSetting

```
public:static void
  enableSetting( Setting setting,
                 bool enable = true )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to enable the IPerformanceSettings::Setting specified by setting.

isSettingEnabled

```
public:static bool isSettingEnabled(Setting setting)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the IPerformanceSettings::Setting specified by setting is enabled.

IPerformanceSettings - Enumerations

Setting

Each of these enumerators specifies one of the options you control using enableSetting or disableSetting.

singleThreaded

When this setting is enabled, it causes IResourceLock to bypass setting and releasing locks. This setting applies to all IResourceLock objects, regardless of whether they are used by the Open Class Library implementation or by client code. Enabling this setting implies singleThreadedGUI. The default for this setting is disabled.

Enabling this setting causes your single threaded application to run faster because the overhead of cross thread serialization is avoided.

singleThreadedGUI

When this setting is enabled, it causes the implementation of IWindow and its derived classes to bypass the setting and releasing of serialization locks. It is similar to singleThreaded, except that its scope is more limited. The default for this setting is disabled in the Windows and OS/2 environments and enabled in AIX for PowerPC.

Enable this setting when your application creates, manipulates, and destroys IWindow objects using a single thread. With this setting enabled, you can still use INonGUIThread, IResourceLock and related classes to manipulate and synchronize threads not related to the user interface.

validateWindowHandle

Enable this setting to cause IWindow::handle to check the validity of the window handle before returning it. IWindow::handle checks the validity of the handle by calling IWindow::isValid. If IWindow::isValid returns false, IWindow::handle throws an exception. The default for this setting is disabled.

dynamicToolBarButtons

Enabling this setting will cause the functions IToolBarButton::setStandardBitmapSize, IToolBarButton::setStandardTextLines and IToolBarButton::setStandardTextWidth to immediately update all

toolbar buttons. Otherwise the standard button settings will be used during the creation of new toolbar buttons.

IPerformanceSettings - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPrimalLock

IPrimalLock is a very specialized resource lock class. It is intended only for use during lazy evaluation of static/global objects. For instance, many developers don't want to create static/global objects because of the order of construction problems involved. They would prefer to provide access methods which dynamically allocate these objects only when they are actually required. IPrimalLock is intended to make such 'lazy evaluation' thread safe.

IPrimalLock is available for use at all times, including dynamic link library (or shared library) initialization and static/global constructor time.

DO NOT create a global or static IPrimalLock object, because that will permanently lock the primal lock and your application will likely hang forever. It is only for use on the stack, to create scope based locking during initialization.

DO NOT use IPrimalLock as a general purpose thread synchronization mechanism. Use IPrimalLock to allocate an IPrivateResource or ISharedResource object in a thread safe way and use that resource object to do your thread synchronization. Otherwise, IPrimalLock would become a choke point causing unwarranted blocking of otherwise unrelated threads.

Class header file: iprimlck.hpp

IPrimalLock - Hierarchy List

IPrimalLock

IPrimalLock - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IPrimalLock

```
public:~IPrimalLock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor will release the primal lock, letting go any other thread that is waiting on it.

IPrimalLock

```
public:IPrimalLock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The constructor will wait until it can obtain the primal lock. Once the lock is obtained, it will return and you can then assume that you have control of the lock, and do work within the scope of the lock that will not be interrupted by other threads.

IPrimalLock - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPrivateCondition

IPrivateCondition represents an unnamed derivative of the abstract base ICondition class. Refer to the parent class for details on conditions and how they are used.

Class header file: iconditn.hpp

IPrivateCondition - Hierarchy List

ICondition

IPrivateCondition

IPrivateCondition - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct private condition objects.

~IPrivateCondition

```
public:virtual ~IPrivateCondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys any internal system services required to implement the condition on each particular platform and destroys this object.

Portability Considerations

The action of destroying a condition on which threads are waiting is undefined and platform specific. The most likely scenario is an 'owner died' error from the underlying operating system, caused by the underlying system services used to block the waiting threads.

IPrivateCondition

```
public:IPrivateCondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a private condition object. Private condition constructors have no required parameters. The new private condition can only be used within the context of the process in which it is created.

Exception

Exception	Description
IConditionInvalid	if there is any failure to create the underlying system resources required to implement the condition.

Waiting and signalling

Use these member functions to wait on a private condition, or signal other threads which are waiting on a private condition.

broadcast

```
public:virtual void broadcast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows all threads, currently waiting on this condition, to return from their wait operations. In some platform implementations, threads entering wait() while the broadcast is being done might also be allowed to pass on through (it does not necessarily take a snapshot of waiting threads).

signal

```
public:virtual void signal()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Signals the condition, which allows one other thread to return from a wait on the condition. Only one thread is allowed through. If no threads are waiting, then this has no effect. Therefore, signal() is a pulse type operation. It does not remain signaled.

wait

Waits for the condition to be signalled or broadcast by another thread of the current process. An optional timeout can be provided, which defaults to an infinite wait if none is explicitly given.

Overload 1

```
public:virtual bool
  wait( IResourceLock& aLock,
        ioc::numeric_t timeOut = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool wait(ioc::numeric_t timeOut = - 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPrivateCondition - Inherited Member Functions and Data

Inherited Public Functions

ICondition			
Definition	**Page Number**	**Definition**	**Page Number**
~ICondition	4	broadcast	4
Relock	4	signal	5
Unlock	4	wait	5

Inherited Public Data

None

Inherited Protected Functions

ICondition			
Definition	**Page Number**	**Definition**	**Page Number**
ICondition	4		

Inherited Protected Data

None

IPrivateResource

The IPrivateResource class defines a resource that is used within a single process.

IResource and its inherited classes represent resources in the user's problem domain. Objects of these classes name a particular resource. For example, a user wants to serialize access to the data in a window list collection. The user creates an IResource inherited class to represent the window list and then uses IResourceLock (preferred) or a function of IResource itself to lock and unlock the resource to serialize the access. You can use this class either alone (lock and unlock) or in combination with an IResourceLock, whose constructor locks the resource and whose destructor unlocks the resource. If you need the lock only for a block of code, use IResourceLock. Use static objects because you only need a single object of a particular IResource.

The static object pointer discussions exist because there is no language-defined way to ensure that static objects are constructed when needed. Also, they are always constructed even if they are never needed. Combining a static pointer to an object with a static function to reference the object and then creating the object, if necessary, defers creating the object until it is needed.

Use this class for resources that are limited to the same process (as opposed to a public resource that is used by more than one process via its name).

You can use this class as a static key to serialize access to a private resource. You can also use this class as a mechanism to ensure that the static resource is constructed prior to being used.

Some basic guidelines include the following:

- Use a static pointer to the resource rather than a static object.

- Always access the resource using a static function rather than accessing it directly with the static pointer.
- When the resource is accessed through the static function, allocate it if the static pointer is 0.
- Provide a new class that represents the static pointers for a particular class or component. This new class does not require a constructor, but it does require a destructor that destroys the static objects used by the component.

Class header file: ireslock.hpp

IPrivateResource - Hierarchy List

IResource

IPrivateResource

IPrivateResource - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IPrivateResource

```
public:virtual ~IPrivateResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys a private resource object.

Exception

Exception	Description
IAccessError	The private resource object was not deleted. The semaphore handle may be invalid.

IPrivateResource

```
public:IPrivateResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a private resource object. The only way to construct objects of this class is with this default constructor.

Exception

Exception	Description
IAccessError	The private resource object was not created. The memory may be exhausted or the semaphore handle limit on your platform may be exceeded.

Handle Functions

Use this member function to obtain the handle to the resource.

handle

```
protected:IPrivateSemaphoreHandle& handle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle for the private resource. This is only available to derived classes.

Lock and Unlock

Use these member functions to lock and unlock the resource.

lock

```
public:virtual IPrivateResource&
  lock( ioc::numeric_t timeOut = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Acquires serialized access to the resource, preventing other threads of execution from accessing it. You can specify the value for **timeout** in milliseconds with -1 indicating an indefinite wait and 0 requesting an immediate return.

Locks can be obtained multiple times on the same resource by the same thread. A corresponding number of unlocks must be performed to release the resource.

timeout

The number of milliseconds the caller is willing to wait to get the lock. If the lock cannot be acquired within that time period, then an exception is thrown.

unlock

```
public:virtual IPrivateResource& unlock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Releases access to the resource, allowing other threads of execution to access it. If multiple locks have been acquired by the same thread, the lock count is decremented. Once the lock count reaches zero, the resource will be released.

Exception

Exception	Description
IAccessError	Unexpected error while accessing underlying system object.

IPrivateResource - Inherited Member Functions and Data

Inherited Public Functions

IResource			
Definition	**Page Number**	**Definition**	**Page Number**
~IResource	111	IResource	111
lock	111	unlock	112

Inherited Public Data

None

Inherited Protected Functions

IResource			
Definition	**Page Number**	**Definition**	**Page Number**
relockAll	112	unlockAll	112

Inherited Protected Data

None

IPrivateSemaphoreHandle

This class represents the handle to a private semaphore, which can be quite different from host to host.

Class header file: ibhandle.hpp

IPrivateSemaphoreHandle - Hierarchy List

IPrivateSemaphoreHandle

IPrivateSemaphoreHandle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct and assign private semaphore handle objects.

~IPrivateSemaphoreHandle

```
public:~IPrivateSemaphoreHandle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the private semaphore handle.

IPrivateSemaphoreHandle

```
public:IPrivateSemaphoreHandle(Value privSemId = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IPrivateSemaphoreHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

operator =

```
public:IPrivateSemaphoreHandle& operator =(Value privSemId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns a new handle value to this object.

Diagnostics

Use these member functions to obtain diagnostic information.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

Member Access

Use these member functions to access the data of this class.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

IPrivateSemaphoreHandle - Type Definitions

Value

A typedef for the host specific type of the semaphore handle member.

Variation 1

```
typedef unsigned long Value
```

This is supported on **OS/2**

Variation 2

```
typedef void * Value
```

This is supported on **WIN AIX OS/400 z/OS**

IPrivateSemaphoreHandle - Associated Globals

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IPrivateSemaphoreHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IPrivateSemaphoreHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPrivateSemaphoreHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IProcessId

This class represents a process id, which is a number that uniquely identifies a running process.

Class header file: ibhandle.hpp

IProcessId - Hierarchy List

IProcessId

IProcessId - Member Functions and Data by Group

Constructors & Destructor

Use this member function to construct process id objects.

IProcessId

```
public:IProcessId(Value pid = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IProcessId object from an operating system handle of type Value. Typically this value originates in a system API call.

Diagnostics

Use these member functions to obtain debug information for process id objects.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

Member Access

Use these member functions to acess the data of this class.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

IProcessId - Type Definitions

Value

A typedef for the host specific type of the process id member.

Variation 1

```
typedef void * Value
```

This is supported on **WIN**

Variation 2

```
typedef unsigned long Value
```

This is supported on **OS/2 AIX OS/400 z/OS**

IProcessId - Associated Globals

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IProcessId& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IProcessId& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IProcessId - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IResource

The IResource class is the abstract base resource class. Use a derived class to identify a serially reusable resource. The concept of resource is similar to the common definition of mutex and monitor. These resources can be limited to the current process, as described by the IPrivateResource class, or they can be shared across multiple processes, as described by the ISharedResource class.

Class header file: ireslock.hpp

IResource - Hierarchy List

IResource

ISharedResource

IPrivateResource

IResource - Member Functions and Data by Group

Constructors & Destructor

You cannot construct or destruct objects of this class. Derive a new class from this class to identify a serially reusable resource if the derived Open Class Library classes do not meet your needs.

~IResource

```
public:virtual ~IResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the resource, releasing it if it is currently locked.

IResource

```
public:IResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a resource object. This is only available to derived classes.

Resource Locking

Use the member functions to acquire or release serialized access to this resource.

lock

```
public:virtual IResource& lock(ioc::numeric_t timeOut = - 1) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Acquires serialized access to the resource, preventing other threads of execution from accessing it. You can specify the value for **timeout** in milliseconds with -1 indicating an indefinite wait and 0 requesting an immediate return.

Locks can be obtained multiple times on the same resource by the same thread. A corresponding number of unlocks must be performed to release the resource.

timeout

The number of milliseconds the caller is willing to wait to get the lock. If the lock cannot be acquired within that time period, then an exception is thrown.

AIX Considerations

The AIX timer services used to implement the resource lock timeout have a resolution of one second. The lock function rounds up the specified timeout value to the next one-second interval.

z/OS Considerations

Timeout values are not supported in a POSIX environment. The default value of -1 is assumed. Resource locks are only supported when the kernel is available and active. In a non-UNIX environment, the lock() function has no effect.

unlock

```
public:virtual IResource& unlock() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Releases access to the resource, allowing other threads of execution to access it. If multiple locks have been acquired by the same thread, the lock count is decremented. Once the lock count reaches zero, the resource will be released.

Exception

Exception	Description
IAccessError	Unexpected error while accessing underlying system object.

z/OS Considerations

Resource locks are only supported when the kernel is available and active. In a non-UNIX environment, the unlock() function has no effect.

relockAll

```
protected:void relockAll(ioc::numeric_t times)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Relocks all locks on the resource object the specified number of times.

unlockAll

```
protected:ioc::numeric_t unlockAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Releases all access to the resource.

IResource - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IResourceId

The IResourceId class binds a numeric resource identifier with an IResourceLibrary object used to load the resource. Typically, you do not specify the resource library so that the IResourceId object binds to the user resource library it obtains by calling IApplication::current().userResourceLibrary. Classes in the Open Class Library that load a resource typically accept an IResourceId as input to describe the resources.

Class header file: ireslib.hpp

IResourceId - Hierarchy List

IResourceId

IResourceId - Member Functions and Data by Group

Constructors & Destructor

You can construct and copy objects of this class. You cannot assign objects of this class because its assignment operator is private. You can construct objects of this class by providing a resource identifier, a resource identifier and a reference to an IResourceLibrary, or a resource identifier and a reference to an IDynamicLinkLibrary. You can also construct or initialize an object of this class from another IResourceId object.

IResourceId

Constructs an IResourceId object.

Overload 1

```
public:IResourceId(const IResourceId& resourceId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an IResourceId object using a reference to an existing IResourceId.

resourceId

Reference to an existing IResourceId object.

Overload 2

```
public:IResourceId(unsigned long resourceId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Identifies a resource in the default library for application-supplied resources. The constructor uses the resource library returned by a call to ICurrentApplication::userResourceLibrary.

resourceId
Unsigned long value that represents the resource identifier.

Overload 3

```
public:IResourceId( unsigned long resourceId,
            const IResourceLibrary& resourceLibrary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an IResourceId object that binds the resource identifier with a reference to an IResourceLibrary. Use this constructor if the resource library is not the one returned by ICurrentApplication::userResourceLibrary..

resourceId
Unsigned long value that represents the resource identifier.

resourceLibrary
Reference to the resource library where the resource resides.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides debugging information about the class object. It returns a string that contains the resource identifier as well as the results of a call to IResourceLibrary::asDebugInfo for its resource library.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides textual information about the class object. It returns the resource identifier.

Resources

Use these member functions to get the resource identifier or a reference to the resource library object for this class.

id

```
public:unsigned long id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the identifier used for the resource.

operator unsigned long

```
public:operator unsigned long() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the identifier used for the resource. Using this operator is the same as calling IResourceId::id.

resourceLibrary

```
public:const IResourceLibrary& resourceLibrary() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a const reference to the resource library.

IResourceId - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IResourceLibrary

The IResourceLibrary class loads resources, such as strings, menus, and bitmaps, from an application's executable file. Use the IDynamicLinkLibrary class to load resources that are stored in a dynamic link library.

Typically, you do not need to use this class as most classes in Open Class Library that require resources, provide functions that accept an IResourceId and load the resources from the resource file themselves.

To obtain either the Open Class Library's resource library or the default user resource library, call ICurrentApplication::resourceLibrary or ICurrentApplication::userResourceLibrary, respectively.

Class header file: ireslib.hpp

IResourceLibrary - Hierarchy List

IResourceLibrary

IDynamicLinkLibrary

IResourceLibrary - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign, and destruct objects of this class. You can construct objects of this class by using the default constructor, which does not accept any parameters, or by using the copy constructor to copy one resource library to another.

~IResourceLibrary

```
public:virtual ~IResourceLibrary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IResourceLibrary object.

IResourceLibrary

Constructs an IResourceLibrary object.

Overload 1

```
public:IResourceLibrary(const IResourceLibrary& resLibrary)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a resource library object using a reference to an existing resource library object.

resLibrary
Reference to an existing resource library object.

Overload 2

```
public:IResourceLibrary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a resource library object using this, the default constructor.

operator =

```
public:IResourceLibrary&
  operator =( const IResourceLibrary& resLibrary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the member data of an object of this class to another object of this class.

resLibrary
Reference to an existing resource library object.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides debugging information about the class object. It returns a string that contains the file name of the resource library.

asString

```
public:virtual IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides textual information about the class object. It returns a string that contains the file name of the resource library.

Resource Information

Use these member functions to query general resource information.

fileName

```
public:virtual IString fileName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the fully qualified file name of the resource library.

handle

```
public:virtual IModuleHandle handle() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle to the resource library that is used to load the resources. IResourceLibrary objects always return 0 as the handle.

OS/2 Considerations

This is always 0 for an IResourceLibrary object.

isOpen

```
public:virtual bool isOpen() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the open state of the resource file. IResourceLibrary objects always return true.

Resource Loading

Use these member functions to load specific resources from a resource file.

loadAccelTable

```
public:virtual IAccelTblHandle
  loadAccelTable( unsigned long accelTableId ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads an accelerator table from a resource file.

accelTableId

Unsigned long value that represents the accelerator table identifier.

Exception

Exception	Description
IAccessError	The accelerator table was not loaded. The accelerator table identifier may be invalid.

loadBitmap

Loads a bitmap from a resource file.

If the **cached** argument is true, the bitmap is reference-counted. Therefore, if multiple requests are made for the bitmap, the reference count is incremented and the same copy of the bitmap is returned. When all of the references to the bitmap are deleted, the Open Class Library deletes the bitmap.

If you set the **cached** argument to false, IResourceLibrary creates a new copy of the bitmap. It is your responsibility to delete bitmaps created in this manner.

Overload 1

```
public:virtual IBitmapHandle
  loadBitmap( unsigned long bitmapId,
              bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a bitmap from a resource file.

bitmapId
Unsigned long value that represents the bitmap identifier.
cached
A boolean value that determines caching.
Exception

Exception	Description
IAccessError	The bitmap was not loaded. The bitmap identifier may be invalid.

Overload 2

```
public:virtual IBitmapHandle
  loadBitmap( unsigned long bitmapId,
              const ISize& bitmapSize,
              bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a bitmap from a resource file. Use this version of the function if you want to specify the size at which the bitmap is loaded.

bitmapId
Unsigned long value that represents the bitmap identifier.

bitmapSize
Reference to a size object that determines at which size the bitmap is loaded.

cached
A boolean value that determines caching.
Exception

Exception	Description
IAccessError	The bitmap was not loaded. The bitmap identifier may be invalid.

loadDialog

```
public:virtual IWindowHandle
  loadDialog( unsigned long dialogId,
              IWindow* dialogParent,
              IWindow* dialogOwner,
              IWinProc* dialogProcedure,
              void* dialogCreateParameters ) const
```

This is supported on **WIN OS/2**

Loads a dialog resource with the specified parent and owner windows.

You normally load dialog windows by creating IFrameWindow objects with an IFrameWindow::FrameSource of IFrameWindow::dialogWindow.
dialogId
Unsigned long value that represents the dialog identifier.
dialogParent
Pointer to a window object that is the dialog's parent.
dialogOwner
Pointer to a window object that is the dialog's owner.

dialogProcedure
Pointer to a dialog procedure for the dialog you are loading.
dialogCreateParameters
Pointer to an application-defined data area. The first 2 bytes of the data referenced by the pointer should be the total size of the data.
Exception

Exception	Description
IAccessError	The dialog was not loaded. The dialog identifier, parent, owner, procedure, or create parameters may be invalid.

AIX Considerations

AIX does not support dialogs.

loadHelpTable

```
public:virtual IResourceLibrary&
  loadHelpTable( IWindow* helpInstance,
                 unsigned long helpTableId ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a help table from a resource file. The IHelpWindow class provides information on help windows.
helpInstance
Pointer to the IWindow object for the help instance.
helpTableId
Unsigned long value that represents the help table identifier.
Exception

Exception	Description
IAccessError	The help table was not loaded. The help table identifier or the help table instance may be invalid.

loadIcon

```
public:virtual IPointerHandle
  loadIcon( unsigned long iconId,
            bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads an icon from a resource file.

If the **cached** argument is true, the icon is reference-counted. Therefore, if multiple requests are made for the icon, the reference count is incremented and the same copy of the icon is used. When all of the references to the icon are removed, the Open Class Library deletes the icon.

If you set the **cached** argument to false, IResourceLibrary creates a new copy of the icon. It is your responsibility to delete icons created in this manner.
iconId Unsigned long value that represents the icon identifier.
cached
A boolean value that determines caching.
Exception

Exception	Description
IAccessError	The icon was not loaded. The icon identifier may be invalid.

loadMenu

Loads a menu resource from a resource file.

Overload 1

```
public:virtual void
  loadMenu( unsigned long menuId,
            const IWindowHandle& menuHandle,
            IMenu* parentMenu ) const
```

This is supported on **AIX**

menuId
Unsigned long value that represents the identifier of the menu resource.

menuHandle
Reference to the handle of the menu window.

parentMenu
Pointer to a menu object that is the menu's parent.

Exception

Exception	Description
IInvalidParameter	The parentMenu was 0.
IAccessError	The menu was not loaded. The menu identifier or the menu handle may be invalid.

Portability Considerations

To achieve portable code, you should use a unique ID value for each item in a given menu. This is because the UNIX implementation for loading menus from a resource menu template requires unique IDs on, for example, each submenu item even though you may not use them yourself.

Overload 2

```
public:virtual IMenuHandle
  loadMenu( unsigned long menuId,
            IWindow* menuOwner ) const
```

This is supported on **WIN OS/2**

menuId
Unsigned long value that represents the menu identifier.

menuOwner
Pointer to a window object that is the menu's owner.

Exception

Exception	Description
IAccessError	The menu was not loaded. The menu identifier or the menu owner may be invalid.

loadMessage

```
public:virtual IString loadMessage(unsigned long messageId) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a message resource from a resource file using the specified message identifier.

messageId
Unsigned long value that represents the message identifier.
Exception

Exception	Description
IAccessError	The message string was not loaded. The message identifier may be invalid.

loadPointer

```
public:virtual IPointerHandle
  loadPointer( unsigned long iconId,
               bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a pointer with the specified resource identifier from a resource file.

If the **cached** argument is true, the pointer is reference-counted. Therefore, if multiple requests are made for the pointer, the reference count is incremented and the same copy of the pointer is used. When all of the references to the pointer are removed, the Open Class Library deletes the icon.

If you set the **cached** argument to false, IResourceLibrary creates a new copy of the pointer. It is your responsibility to delete pointers created in this manner.

Calling this function is the same as calling IResourceLibrary::loadIcon.
iconId Unsigned long value that represents the icon identifier.
cached
A boolean value that determines caching.
Exception

Exception	Description
IAccessError	The pointer was not loaded. The icon identifier may be invalid.

Portability Considerations

On the OS/2 and Windows operating systems, this function is the same as calling loadIcon. On Motif, the pointer created via this function is a single-plane image which is suitable for use as a cursor.

loadString

```
public:virtual IString loadString(unsigned long stringId) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Loads a string resource from a resource file using the specified string identifier.

stringId
Unsigned long value that represents the string identifier.

Exception

Exception	Description
IAccessError	The string was not loaded. The string identifier may be invalid.

tryToLoadBitmap

Attempts to load a bitmap from a resource file. If this function cannot load the bitmap, it returns an IBitmapHandle with a 0 handle value rather than throwing an exception.

If the **cached** argument is true, the bitmap is reference-counted. Therefore, if multiple requests are made for the bitmap, the reference count is incremented and the same copy of the bitmap is used. When all of the references to the bitmap are deleted, the Open Class Library deletes the bitmap.

If you set the **cached** argument to false, IResourceLibrary creates a new copy of the bitmap. It is your responsibility to delete bitmaps created in this manner.

Overload 1

```
public:virtual IBitmapHandle
  tryToLoadBitmap( unsigned long bitmapId,
                   const ISize& bitmapSize,
                   bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of the function if you want to specify the size at which the bitmap is loaded.

bitmapId
Unsigned long value that represents the bitmap identifier.

bitmapSize
Reference to a size object that determines at which size the bitmap is loaded.

cached
A boolean value that determines caching.

Overload 2

```
public:virtual IBitmapHandle
  tryToLoadBitmap( unsigned long bitmapId,
                   bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

bitmapId
Unsigned long value that represents the bitmap identifier.

cached
A boolean value that determines caching.

tryToLoadIcon

```
public:virtual IPointerHandle
  tryToLoadIcon( unsigned long iconId,
                 bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to load an icon from a resource file. If this function cannot load the icon, it returns an IPointerHandle with a 0 handle value rather than throwing an exception.

If the **cached** argument is true, the icon is reference-counted. Therefore, if multiple requests are made for the icon, the reference count is incremented and the same copy of the icon is used. When all of the references to the icon are deleted, the Open Class Library deletes the icon.

If you set the **cached** argument to false, IResourceLibrary creates a new copy of the icon. It is your responsibility to delete icons created in this manner.

iconId Unsigned long value that represents the icon identifier.

cached

A boolean value that determines caching.

tryToLoadMessage

```
public:virtual IString
  tryToLoadMessage( unsigned long messageId ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to load a message resource from a resource file. If this function cannot load the message, it returns an empty IString rather than throwing an exception.

messageId

Unsigned long value that represents the message identifier.

tryToLoadPointer

```
public:virtual IPointerHandle
  tryToLoadPointer( unsigned long iconId,
                    bool cached = true ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to load a pointer from a resource file. If this function cannot load the icon, it returns an IPointerHandle with a 0 handle value rather than throwing an exception.

If the **cached** argument is true, the icon is reference-counted. Therefore, if multiple requests are made for the icon, the reference count is incremented and the same copy of the icon is used. When all of the references to the icon are deleted, the Open Class Library deletes the icon.

iconId Unsigned long value that represents the icon identifier.

cached

A boolean value that determines caching.

tryToLoadString

```
public:virtual IString
  tryToLoadString( unsigned long stringId ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to load a string resource from a resource file. If this function cannot load the string, it returns an empty IString rather than throwing an exception.

stringId

Unsigned long value that represents the string identifier.

resourceAddress

```
protected:virtual void* resourceAddress() const
```

This is supported on **AIX**

This is an internal use only function used to load resources on the Motif platform.

IResourceLibrary - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IResourceLock

The IResourceLock class locks a resource for a specified period of time.

A simple yet effective use of this class is to declare a local object of this class that is in scope for the period of time that the resource needs to be locked. The lock is automatically removed when the block of code is exited, thereby forcing the object out of scope.

If the specified period of time is reached before the resource can be acquired, an IResourceExhausted exception is thrown.

Class header file: ireslock.hpp

IResourceLock - Hierarchy List

IResourceLock

IResourceLock - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IResourceLock

```
public:virtual ~IResourceLock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the resource lock and releases the lock that was obtained by the constructor.

IResourceLock

```
public:IResourceLock( IResource& resource,
                      ioc::numeric_t timeOut = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a resource lock object.

resource

Reference to the resource being locked.

timeout

The number of milliseconds the caller is willing to wait to get the lock. If the lock cannot be acquired within that time period, then an exception is thrown. The default value, -1, is an indefinite wait.

AIX Considerations

The AIX timer services used to implement the resource lock timeout have a resolution of one second. The lock function rounds up the specified timeout value to the next one-second interval.

z/OS Considerations

Timeout values are not supported in a POSIX environment. The default value of -1 is assumed. Resource locks are only supported when the kernel is available and active.

Lock and Unlock Functions

Use these member functions to lock and unlock the resource.

clearLock

```
protected:virtual IResourceLock& clearLock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Releases the resource. The destructor calls this method to unlock the resource.

Exception

Exception	Description
IAccessError	Unexpected error while accessing underlying system object.

setLock

```
protected:virtual IResourceLock&
  setLock( ioc::numeric_t timeOut = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Acquires the resource. You can specify the value for **timeOut** in milliseconds with -1, indicating an indefinite wait, and 0, requesting an immediate return.

The constructor calls this function to lock the resource.

timeOut

The number of milliseconds the caller is willing to wait to get the lock. If the lock cannot be acquired within this time, then an exception is thrown. It defaults to -1 which means to wait forever.

IResourceLock - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ISharedCondition

Shared conditions are named versions of the base ICondition class. Refer to the parent class for a description of conditions and how they are used.

Class header file: iconditn.hpp

ISharedCondition - Hierarchy List

ICondition

ISharedCondition

ISharedCondition - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct shared condition objects.

~ISharedCondition

```
public:virtual ~ISharedCondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor destroys the platform specific mechanisms which are used to implement the condition; however, since these mechanisms are also named and shared, they are not actually closed until all users of them have destroyed their copies.

ISharedCondition

Constructs a shared condition object. A shared condition may be used by multiple processes, as long as those processes know the name of the condition.

If the condition already exists, it is opened; else, it is created if the name is valid.

Overload 1

```
public:ISharedCondition()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a shared condition object with a uniquely generated name. In order to be used by other processes, this name would have to be transmitted to them by some other inter-process communication (IPC) mechanism.

Exception

Exception	Description
IConditionInvalid	if there is any failure to create the underlying system resources required to implement the condition.

Overload 2

```
public:ISharedCondition(const char* keyName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a shared condition with the specified name. Any other processes which know this name can open this shared condition and use it.

keyName

The name to give to the shared condition object. The default constructor does not accept a name, and will build a unique name for you. This name may subsequently be queried and passed to other processes via some other inter-process communication (IPC) mechanism if desired.

Windows Considerations

Terminal Services: Windows Terminal Server programs have two name spaces available for system objects: the session name space and the global name space. You can explicitly specify whether this constructor should create its system object in the global or session name space through the name you pass to the constructor. Prefix the name with ″Global\″ to create the system object in the global name space. Prefix the name with ″Local\″ to create the system object in the global name space.

For example,

ISharedCondition(″Global\\keyName″)
will create the name in the global name space

ISharedCondition(″Local\\keyName″)
will create the name in session name space

ISharedCondition(″keyName″)
will create the name in session which is the default

Windows 2000: On Windows 2000 systems without Terminal Services running, the ″Global\″ and ″Local\″ prefixes are ignored.

Member Access

Use these member functions to access the data of a shared condition object.

keyName

```
public:IString keyName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the key name for this condition object. Note that this is the name that you gave, but this is not necessarily the host specific name that actually gets used internally. The naming scheme of the host operating system often requires some ornamentation.

Static methods

These member functions perform static operations on shared conditions.

exists

```
public:static bool exists(const char* keyName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks to see whether the named condition exists. Note that, as with any such shared resource in a multitasking environment, you cannot always

assume that it will not come into existence after this call but before you can do something based on the result. Therefore, only consider it advisory.

keyName

The name to check for. If a condition by this name exists, true is returned.

Waiting and signalling

Use these member functions to wait on a shared condition or signal other threads which are waiting on a shared condition.

broadcast

```
public:virtual void broadcast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows all threads, currently waiting on this condition, to return from their wait operations. In some platform implementations, threads entering wait() while the broadcast is being done might also be allowed to pass on through (it does not take a snapshot of waiting threads necessarily).

signal

```
public:virtual void signal()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Signals the condition, which allows one waiting thread to return from a wait on the condition. Only one thread is allowed through. If no threads are waiting, then this has no effect. Therefore, signal() is a pulse type operation. It does not remain signalled.

Exception

Exception	Description
IConditionInvalid	if any errors occur in the host APIs required to perform the signaling operation.

wait

Waits for the condition to be signalled or broadcast by another thread or process. An optional timeout can be provided, which defaults to an infinite wait if none is explicitly given.

Overload 1

```
public:virtual bool wait(ioc::numeric_t timeOut = - 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool
  wait( IResourceLock& alock,
        ioc::numeric_t timeOut = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISharedCondition - Inherited Member Functions and Data

Inherited Public Functions

ICondition			
Definition	**Page Number**	**Definition**	**Page Number**
~ICondition	4	broadcast	4
Relock	4	signal	5
Unlock	4	wait	5

Inherited Public Data

None

Inherited Protected Functions

ICondition			
Definition	**Page Number**	**Definition**	**Page Number**
ICondition	4		

Inherited Protected Data

None

ISharedResource

The ISharedResource class defines a resource that can be shared across multiple processes. Use the name you provide on construction to distinguish between shared resources.

Portability Considerations

Use a keyName value that is a valid file name in all of the target environments.

AIX Considerations

You might use ISharedResource objects to serialize resource access with applications that were not written using the Open Class Library. To make this work correctly, your non-Open Class Library application needs to use the AIX semaphore object in a manner consistent with the Open Class Library. To help you understand how the semaphore object is used by the Open Class Library, the basic interaction with the system's subroutines is described below.

The AIX semop subroutine is used for the semaphore lock and release operations. The ISharedResource constructor uses the semget subroutine to allocate a set containing two semaphores. The first semaphore in this set represents the state of the lock. The constructor initializes the semval field to free (value of 0). A lock operation increments the value, while an unlock operation decrements the value.

Recursive lock operations by the same process are allowed. In this case, the resource remains owned by the process until an equal number of unlock operations are done or the object is deleted. The second semaphore in the set is used as a reference counter to indicate the number of ISharedResource objects that are referencing this semaphore set. The

constructor increments this counter and the destructor decrements it. When the counter reaches 0, the destructor calls the semctl subroutine to delete the semaphore set.

The constructor determines the semaphore identifier to use in the semget subroutine by using the ftok system subroutine. If the parameter provided on the constructor begins with a slash (/) or is an existing file, the parameter is used as a file name for the ftok call and is the basis of the semaphore identifier token. If the parameter is not the name of an existing file, the constructor interprets the string as the name of a file in the /tmp directory. If this file does not exist, it is created. You can use keyName to obtain the file name used on the ftok subroutine call. The constructor always uses a value of 1 for the second parameter of ftok.

z/OS Considerations

The ISharedResource class is only available if the kernel is available and active. In a non-UNIX environment, any attempts to instantiate this class will result in an exception of type InvalidRequest being raised.

Class header file: ireslock.hpp

ISharedResource - Hierarchy List

IResource

ISharedResource

ISharedResource - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~ISharedResource

```
public:virtual ~ISharedResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys a shared resource object.

Exception

Exception	Description
IAccessError	The shared resource object was not deleted. The semaphore handle may be invalid.

ISharedResource

```
public:ISharedResource(const char* keyName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a shared resource. You can only construct objects of this class by providing a name that uniquely identifies the resource to be shared.

keyName

Pointer to the name that uniquely identifies the resource.

Exception

Exception	Description
IAccessError	The shared resource object was not created. The memory may be exhausted, the handle limit on your platform may be exceeded, or the **keyName** may be invalid.

Windows Considerations

Terminal Services: Windows Terminal Server programs have two name spaces available for system objects: the session name space and the global name space. You can explicitly specify whether this constructor should create its system object in the global or session name space through the name you pass to the constructor. Prefix the name with ″Global\″ to create the system object in the global name space. Prefix the name with ″Local\″ to create the system object in the global name space.

For example,

ISharedResource(″Global\\keyName″)
will create the name in the global name space

ISharedResource(″Local\\keyName″)
will create the name in session name space

ISharedResource(″keyName″)
will create the name in session which is the default

Windows 2000: On Windows 2000 systems without Terminal Services running, the ″Global\″ and ″Local\″ prefixes are ignored.

Handle Functions

Use this member function to obtain the handle to the resource.

handle

```
protected:ISharedSemaphoreHandle& handle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle for the shared resource. This is only available to derived classes.

Lock and Unlock

Use these member functions to lock and unlock the resource.

lock

```
public:virtual ISharedResource& lock(ioc::numeric_t timeOut = - 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Acquires serialized access to the resource, preventing other threads of execution from accessing it. You can specify the value for **timeout** in milliseconds with -1 indicating an indefinite wait and 0 requesting an immediate return.

Locks can be obtained multiple times on the same resource by the same thread. A corresponding number of unlocks must be performed to release the resource.

timeout

The number of milliseconds the caller is willing to wait to get the lock. If the lock cannot be acquired within that time period, then an exception is thrown.

unlock

```
public:virtual ISharedResource& unlock()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Releases access to the resource, allowing other threads of execution to access it. If multiple locks have been acquired by the same thread, the lock count is decremented. Once the lock count reaches zero, the resource will be released.

Exception

Exception	Description
IAccessError	Unexpected error while accessing underlying system object.

Resource Information

Use these member functions to access shared resource information.

keyName

```
public:IString keyName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name used to identify this shared resource.

ISharedResource - Inherited Member Functions and Data

Inherited Public Functions

IResource			
Definition	**Page Number**	**Definition**	**Page Number**
~IResource	111	IResource	111
lock	111	unlock	112

Inherited Public Data

None

Inherited Protected Functions

IResource			
Definition	**Page Number**	**Definition**	**Page Number**
relockAll	112	unlockAll	112

Inherited Protected Data

None

ISharedSemaphoreHandle

The ISharedSemaphoreHandle class is a wrapper for operating system semaphore handles. An example of a semaphore is a flag in a multiple-user application that prevents simultaneous access to a file.

Class header file: ibhandle.hpp

ISharedSemaphoreHandle - Hierarchy List

ISharedSemaphoreHandle

ISharedSemaphoreHandle - Member Functions and Data by Group

Constructors & Destructor

Use this constructor to create an ISharedSemaphoreHandle object.

ISharedSemaphoreHandle

```
public:ISharedSemaphoreHandle(Value hsem = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ISharedSemaphoreHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

Diagnostics

Use these member functions to obtain diagnostic information about the handle.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

Member Access

Use these member functions to access data of this class.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle value as an unsigned long value.

operator Value

```
public:operator Value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

ISharedSemaphoreHandle - Type Definitions

Value

This type definition defines Value to be a void* on the Windows platforms. On all other platforms Value is defined to be an unsigned long.

Variation 1

```
typedef unsigned long Value
```

This is supported on **OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef void * Value
```

This is supported on **WIN**

ISharedSemaphoreHandle - Associated Globals

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const ISharedSemaphoreHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const ISharedSemaphoreHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISharedSemaphoreHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStandardNotifier

The IStandardNotifier class provides a direct implementation of the notification protocol in the INotifier class.

You can implement a notification protocol in the following way:

- Derive a class from the IStandardNotifier class, which inherits from INotifier, for a direct implementation of the INotifier protocol.
- Derive from the INotifier class and implement your own notification protocol.

Because IWindow inherits from and implements the INotifier protocol, IWindow provides a visual notification implementation. IStandardNotifier inherits from INotifier and can be used for any generic notifier, without the visual interface available in IWindow objects. You might want to derive your classes from IStandardNotifier if you are providing a nonvisual notifier.

Class header file: istdntfy.hpp

IStandardNotifier - Hierarchy List

INotifier

IStandardNotifier

IVCollectionImpl

IVCollection

IStandardNotifier - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, destruct and copy objects of this class.

~IStandardNotifier

```
public:virtual ~IStandardNotifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructing an IStandardNotifier will cause a deleteId notification to be sent to any registered observers. Therefore, it is important that any observers that no longer exist have unregistered themselves from this notifier. Otherwise an error will occur when this notifier tries to notify a non-existent observer. The observer list is deleted.

IStandardNotifier

Constructs an IStandardNotifier object.

Overload 1

```
public:IStandardNotifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor. By default, IStandardNotifier objects are created in a disabled state and therefore require a call to enableNotification before they will pass notifications on to observer objects.

Overload 2

```
public:IStandardNotifier(const IStandardNotifier& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IStandardNotifier object using a copy of an existing IStandardNotifier object. The observer list will be initialized to 0 and will not be copied.

Assignment

Use these member functions to assign one IStandardNotifier object to another.

operator =

```
public:IStandardNotifier&
  operator =( const IStandardNotifier& aStandardNotifier )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the contents of one notifier object to another. The observer list will be initialized to 0 and will not be assigned.

Notification Members

Use these member functions to affect the ability of a part to notify observers of events of interest.

disableNotification

```
public:virtual IStandardNotifier& disableNotification()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the object from sending notifications to registered observers.

enableNotification

```
public:virtual IStandardNotifier&
  enableNotification( bool enable = true )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts the sending of notifications to observers.

isEnabledForNotification

```
public:virtual bool isEnabledForNotification() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if an object is sending notifications to its observers.

notifierAddress

```
public:virtual INotifierAddress notifierAddress() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the notifier address which identifies this notifier.

Observer Addition and Removal

Use these member functions to manage the collection of observers maintained by the notifier.

addObserver

```
protected:virtual IStandardNotifier&
  addObserver( IObserver& observer,
               const IInterest& interest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds an observer to the object's list of observers.

removeAllObservers

```
protected:virtual IStandardNotifier& removeAllObservers()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all observers from the object's observer list.

removeObserver

Removes an observer from the object's observer list.

Overload 1

```
protected:virtual IStandardNotifier&
  removeObserver( IObserver& observer,
                  const IInterest& interest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes an observer from the notifier's list of observers.

observer
Reference to the IObserver for which interest is to be removed..

interest
Reference to the IInterest to be removed from the observer.

Overload 2

```
protected:virtual IStandardNotifier&
  removeObserver( IObserver& observer )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes an observer from the notifier's list of observers.

observer
Reference to the IObserver to be removed from the list of observers.

Observer Notification

These member functions notify observers of a change in a notifier.

notifyObservers

Notifies all observers interested in the notification event.

Note: A public and a protected version of notifyObservers are provided for convenience. The protected version does not require the caller to construct an INotificationEvent to call it. In this case, the construction of the INotificationEvent object occurs in the code of the protected notifyObservers function.

Overload 1

```
public:virtual IStandardNotifier&
  notifyObservers( const INotificationEvent& anEvent )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers interested in a particular notification event.

event Reference to the INotificationEvent to be sent to the observers.

Overload 2

```
protected:virtual IStandardNotifier&
  notifyObservers( const INotificationId& nId )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers interested in a particular notification event.

id Reference to the notification identifier to be sent to the observers in the form of an INotificationEvent.

notifyObserversAsync

```
public:virtual IStandardNotifier&
  notifyObserversAsync( const INotificationEvent& anEvent )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Asynchronously notifies all observers interested in the notification event. Each registered observer must live in a thread which supports asynchronous notifications.

Obsolete

These member functions will soon be obsolete.

observerList

```
protected:virtual IObserverList&
  observerList( const IInterest* anInterest = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the list of IObservers. The list is created it if it does not exist.

IStandardNotifier - Inherited Member Functions and Data

Inherited Public Functions

INotifier			
Definition	**Page Number**	**Definition**	**Page Number**
~INotifier	79	disableNotification	80
enableNotification	80	INotifier	79
isEnabledForNotification	80	notifyObservers	81
notifyObserversAsync	81	threadId	82

Inherited Public Data

INotifier			
Definition	**Page Number**	**Definition**	**Page Number**
allChangesId	79	deleteId	79

Inherited Protected Functions

INotifier			
Definition	**Page Number**	**Definition**	**Page Number**
addObserver	80	notifyObservers	81
observerList	82	removeAllObservers	80
removeObserver	80		

Inherited Protected Data

None

IStdDeque

A wrapper class for IDeque. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdDeque - Hierarchy List

IStandardContainer

IStdDeque

IStdDeque - Member Functions and Data by Group

Miscellaneous Members

IStdDeque

```
public:IStdDeque(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdDeque - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::deque < Element > >
        Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdDeque - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdEqualitySequence

A wrapper class for IEqualitySequence. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdEqualitySequence - Hierarchy List

IStandardContainer

IStdEqualitySequence

IStdEqualitySequence - Member Functions and Data by Group

Miscellaneous Members

IStdEqualitySequence

```
public:IStdEqualitySequence(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdEqualitySequence - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::deque < Element > >
        Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdKeySet

A wrapper class for IKeySet. This class is for internal use only and is not supported.

Class header file: istdkcnr.hpp

IStdKeySet - Hierarchy List

IStandardKeyContainer

IStdKeySet

IStdKeySet - Member Functions and Data by Group

Miscellaneous Members

IStdKeySet

```
public:IStdKeySet( ioc::unumeric_t n,
            Key const & ( * f ) ( Element const & ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySet - Type Definitions

Inherited

```
typedef IStandardKeyContainer < Key , Element , typename std::map
        < Key , Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySet - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdKeySortedSet

A wrapper class for IKeySortedSet. This class is for internal use only and is not supported.

Class header file: istdkcnr.hpp

IStdKeySortedSet - Hierarchy List

IStandardKeyContainer

IStdKeySortedSet

IStdKeySortedSet - Member Functions and Data by Group

Miscellaneous Members

IStdKeySortedSet

```
public:IStdKeySortedSet( ioc::unumeric_t n,
                  Key const & ( * f ) ( Element const & ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSet - Type Definitions

Inherited

```
typedef IStandardKeyContainer < Key , Element , typename std::map
        < Key , Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdKeySortedSetAsTable

A wrapper class for IKeySortedSetAsTable. This class is for internal use only and is not supported.

Class header file: istdkcnr.hpp

IStdKeySortedSetAsTable - Hierarchy List

IStandardKeyContainer

IStdKeySortedSetAsTable

IStdKeySortedSetAsTable - Member Functions and Data by Group

Miscellaneous Members

IStdKeySortedSetAsTable

```
public:IStdKeySortedSetAsTable( ioc::unumeric_t n,
                          Key const & ( * f ) ( Element const & ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSetAsTable - Type Definitions

Inherited

```
typedef IStandardKeyContainer < Key , Element , typename std::map
        < Key , Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdKeySortedSetOnSortedTabularSequence

A wrapper class for IKeySortedSetOnSortedTabularSequence. This class is for internal use only and is not supported.

Class header file: istdkcnr.hpp

IStdKeySortedSetOnSortedTabularSequence - Hierarchy List

IStandardKeyContainer

IStdKeySortedSetOnSortedTabularSequence

IStdKeySortedSetOnSortedTabularSequence - Member Functions and Data by Group

Miscellaneous Members

IStdKeySortedSetOnSortedTabularSequence

```
public:IStdKeySortedSetOnSortedTabularSequence( ioc::unumeric_t n,
                                                Key const & ( * f ) ( Element const & ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSetOnSortedTabularSequence - Type Definitions

Inherited

```
typedef IStandardKeyContainer < Key , Element , typename std::map
        < Key , Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdKeySortedSetOnSortedTabularSequence - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdQueue

A wrapper class for IQueue. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdQueue - Hierarchy List

IStandardContainer

IStdQueue

IStdQueue - Member Functions and Data by Group

Miscellaneous Members

IStdQueue

```
public:IStdQueue(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdQueue - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::deque
        < Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdQueue - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdSequence

A wrapper class for ISequence. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdSequence - Hierarchy List

IStandardContainer

IStdSequence

IStdSequence - Member Functions and Data by Group

Miscellaneous Members

IStdSequence

```
public:IStdSequence(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSequence - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::deque < Element
        > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSequence - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdSet

A wrapper class for ISet. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdSet - Hierarchy List

IStandardContainer

IStdSet

IStdSet - Member Functions and Data by Group

Miscellaneous Members

IStdSet

```
public:IStdSet(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSet - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::set < Element
        > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSet - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdSortedSet

A wrapper class for ISortedSet. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdSortedSet - Hierarchy List

IStandardContainer

IStdSortedSet

IStdSortedSet - Member Functions and Data by Group

Miscellaneous Members

IStdSortedSet

```
public:IStdSortedSet(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSortedSet - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::set < Element >
        > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdSortedSet - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdStack

A wrapper class for IStack. This class is for internal use only and is not supported.

Class header file: istdcnr.hpp

IStdStack - Hierarchy List

IStandardContainer

IStdStack

IStdStack - Member Functions and Data by Group

Miscellaneous Members

IStdStack

```
public:IStdStack(ioc::unumeric_t n = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pop

```
public:void pop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

push

```
public:void push(Element const& e)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

top

```
public:Element const& top() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdStack - Type Definitions

Inherited

```
typedef IStandardContainer < Element , typename std::deque < Element > > Inherited
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStdStack - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ISynchronized

ISynchronized is a template class that is designed to make atomic get, set, and swap operations typesafe and convenient. Most operating systems (or CPUs) offer a mechanism by which a pointer or integral value can be read, written, or exchanged in such a way that they are guaranteed not to be interrupted by another thread, even on a multiple CPU machine.

In general, you will want to use higher level synchronization mechanism, such as the IResource or ICondition family of classes to synchronize multiple threaded code; however, you sometimes will require a way to insure thread safety for some sort of primitive type operation. ISynchronized provides that support.

The operating system or CPU support provided for these operations is inherently typeless, and a little complex sometimes when the values to be exchanged, retrieved, or set are themselves pointers. ISynchronized, can easily handle all of the typelessness for you. It casts incoming and returned values so that you don't have to worry about those issues.

Each ISynchronized object contains a data member, fAtom, which is an instance of the template specialization type. This is the member that is operated on by all of

the atomic operation methods. This data member is guaranteed to have a value of 0 when the object is constructed. The template specialization type must be a 32-bit aligned primitive data type.

AIX Considerations

In 64-bit mode the template specialization type must be a 64-bit aligned primitive data type.

OS/400 Considerations

The template specialization type can be a 16-bit, 32-bit, 64-bit, or 128-bit aligned primitive data type.

Class header file: isynchr.hpp

ISynchronized - Hierarchy List

ISynchronized

ISynchronized - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to contruct objects of this class.

ISynchronized

```
public:ISynchronized()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Member Access

Use these member functions to access data in the synchronized object.

get

```
public:AType get() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Atomically retrieves the current value of the fAtom data member. Keep in mind that, in a multi-threaded environment, this returned value is only advisory. It could be changed by another thread at any time. The fact that ISynchronized prevents multiple access to a data field does not mean you do not have to control your threads' access to the data.

Return

The current value of the object pointed to by the fAtom member.

set

```
public:void set(AType newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the value of the fAtom member atomically. This is an unconditional set of the member and the previous value is just overwritten.

newValue

The new value to set. It will replace the current value.

setConditional

```
public:int setConditional(AType test, AType newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Conditionally sets the fAtom member atomically, if the passed test value equals the current value of the fAtom member.

test The value to test against the current value, to see if the swap should occur.

newValue
The new value to set if the test value matches the current value

Return
false if the new value was set, else true.

Operators

Provides access to the data object being synchronized.

operator AType

```
public:operator AType() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

An implicit conversion operator to extract the value of the fAtom member when the ISynchronized object is used. This allows you to pass the ISynchronized object when a value of the template specialization class is required. This operator does the same thing as get().

Return
The current value of the object pointed to by fAtom

Public Data Members

Public data members.

fAtom

```
public:AType fAtom
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The data member that is operated upon by the atomic operations. It is public for normal C style static initialization.

fFlag

```
public:int fFlag
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISynchronized - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IThreadFn

The IThreadFn class represents functions to be dispatched on secondary threads of execution when you start an INonGUIThread or IThread object. The IThreadFn object is reference counted. As a result, you must create IThreadFn objects using operator new, and Open Class Library will delete the thread function object when it is no longer needed. This is also true for all classes derived from this class.

This class is an abstract thread function class.

Class header file: ingthrd.hpp

IThreadFn - Hierarchy List

IMRefCounted

IThreadFn

IThreadMemberFn

IThreadFn - Member Functions and Data by Group

Constructors & Destructor

You cannot construct or destruct objects of this class because it is an abstract class. Use the template class IThreadMemberFn for dispatching C++ member functions to an object on a new thread.

~IThreadFn

```
public:virtual ~IThreadFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IThreadFn object.

IThreadFn

```
public:IThreadFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IThreadFn object.

Run Function

Use run member function to implement the code that you need to run on secondary threads of execution.

run

```
public:virtual void run() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called when the thread function object is called. Override this function to implement the code that you need to run on secondary threads of execution.

IThreadFn - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

IThreadHandle

This class represents a thread handle, which is a means for interacting with a thread.

Class header file: ibhandle.hpp

IThreadHandle - Hierarchy List

IThreadHandle

IThreadHandle - Member Functions and Data by Group

Constructors & Destructor

Use this member function to construct thread handle objects.

IThreadHandle

```
public:IThreadHandle(Value thread = 0)
```

This is supported on **WIN**

Constructs an IThreadHandle object from an operating system handle of type Value. Typically this value originates in a system API call.

Diagnostics

Use these member functions to obtain debug information about thread handle objects.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN**

Returns the handle as a string of form nnnn.

Member Access

Use these member functions to access data of this class.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN**

Returns the handle value as an unsigned long value.

operator Value

```
public:operator Value() const
```

This is supported on **WIN**

Returns the handle as its operating system dependent type. Use this operator when you use the handle object as an argument to an operating system function.

Public Data

Public data provided by this class.

noHandle

```
public:static const IThreadHandle noHandle
```

This is supported on **WIN**

A handle object that represents the value of an invalid handle.

IThreadHandle - Type Definitions

Value

```
typedef void * Value
```

This is supported on **WIN**

A typedef for the host specific type for the handle member.

IThreadHandle - Associated Globals

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IThreadHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IThreadHandle& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN**

IThreadHandle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IThreadId

This class represents a thread id, which is a unique number that identifies a running thread.

POSIX Considerations

INonGUIThread creates threads using Posix thread APIs. You can use IThreadId::asPthread to access the pthread_t values of these threads.

Class header file: ibhandle.hpp

IThreadId - Hierarchy List

IThreadId

IThreadId - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IThreadId

```
public:~IThreadId()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the thread id.

IThreadId

Constructs an IThreadId object.

Overload 1

```
public:IThreadId()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an empty IThreadId object.

Overload 2

```
public:IThreadId(const pthread_t& thid)
```

This is supported on **OS/400 z/OS**

Construct an IThreadId object from a pthread_t representing an existing Posix thread.

Overload 3

```
public:IThreadId(unsigned long thid)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an IThreadId object from an unsigned long identifier of an existing thread.

Overload 4

```
public:IThreadId(const IThreadId& tid)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs an IThreadId object.

Assignment Operator

The assignment operators for this class.

operator =

```
public:IThreadId& operator =(const IThreadId& tid)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns another id value to this thread id object.

Diagnostics

Use these member functions to obtain diagnostic information for thread id objects.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string containing diagnostic information.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the handle as a string of form nnnn.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX z/OS**

Returns the handle as an unsigned long value.

asUnsignedLong

```
public:unsigned long long asUnsignedLong() const
```

This is supported on **OS/400**

Returns the handle as an unsigned long long value.

isValid

```
public:bool isValid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a boolean value that indicates whether the id is valid.

operator unsigned long

```
public:operator unsigned long() const
```

This is supported on **WIN OS/2 AIX z/OS**

Returns the handle as an unsigned long.

operator unsigned long long

```
public:operator unsigned long long() const
```

This is supported on **OS/400**

Returns the handle as an unsigned long long.

Member Access

Use these member functions to access data of this class.

asPthread

```
public:pthread_t asPthread() const
```

This is supported on **AIX OS/400 z/OS**

Returns the handle as a pthread_t type.

operator pthread_t

```
public:operator pthread_t() const
```

This is supported on **OS/400 z/OS**

Returns the handle as a pthread_t type.

IThreadId - Associated Globals

operator !=

```
bool
  operator !=( const IThreadId& threadId1,
               const IThreadId& threadId2 )
```

Compares two thread id objects for inequality.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IThreadId& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IThreadId& aHandle )
```

Formats this enumeration object to the passed ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IThreadId& threadId1,
               const IThreadId& threadId2 )
```

Compares two thread id objects for equality.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IThreadId - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IThreadLocalBase

IThreadLocalBase is an abstract base class, from which the IThreadLocalStorage template class is derived. IThreadLocalStorage is the class that you should use for per-thread data storage. IThreadLocalBase provides some common functions and manages the magic internal data structures for tracking per-thread data objects.

Class header file: iperthr.hpp

IThreadLocalBase - Hierarchy List

IThreadLocalBase

IThreadLocalStorage

IThreadLocalBase - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IThreadLocalBase

```
protected:virtual ~IThreadLocalBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the thread local base object.

IThreadLocalBase

```
protected:IThreadLocalBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IThreadLocalBase object.

The default constructor sets the **next** field to zero.

Slot Management Methods

These member functions are internal implementation methods for managing the per thread data slots used by this class to store data.

expandThreadSlots

```
protected:void expandThreadSlots(const ioc::unumeric_t newcount)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Expands the number of 'slots' for the calling thread. The number of slots is the number of pointers in the internal list maintained by this class. Each thread local storage object requires a slot for each thread that exists. To avoid excessive expansion, the slot list is actually expanded by an increment value, not just up to the passed count necessarily.

newcount
The new number of slots needed. This is only advisory and the actual slot count is rounded up to the next internal expansion increment.

getNextSlot

```
protected:ioc::unumeric_t getNextSlot()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Used internally for slot management. It will find the next unused slot on the calling thread, expanding the slot list if that is required. It does any required synchronization to ensure that global resources are accessed in a thread safe way.

getSlot

```
protected:void* getSlot(const ioc::unumeric_t index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the pointer value in the indicated slot for the calling thread. Note that it does not actually expand the slot list, since it can just return zero if that slot does not yet exist for this thread. Only the setting of a slot requires that the list actually be expanded. Each thread local object stores its index to the slot list, so the value passed is that stored index.

index The index of the slot (of the calling thread) to get the pointer from.

setSlot

```
protected:void setSlot(const ioc::unumeric_t index, void* data)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the passed data pointer into the indicated per thread data slot.

index The index of the slot for the calling thread that is to be set.

data The data pointer to put into the indicated slot.

Storage Management Methods

These member functions manage the data pointer stored in thread local storage objects.

freeAllStorage

```
public:static void freeAllStorage()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Frees all of the data associated with a thread. It should be called as the very last thing that a thread does, otherwise it might free memory before other code called on that thread is done with it.

freeStorage

```
protected:virtual void freeStorage() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A virtual method that each instantiation of the derived class, IThreadLocalStorage, overrides. Since the type of the data object stored is only known by the derived template class, this method cannot be implemented at the IThreadLocalBase level. However, internally, storage can be freed via the base class, which allows the internal implementation to maintain a list of all thread local storage objects by way of the IThreadLocalBase interface and still be able to clean up the associated data.

IThreadLocalBase - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IThreadLocalBase			
Definition	**Page Number**	**Definition**	**Page Number**
`IThreadLocalBase`	156	`operator =`	156

Inherited Protected Data

None

IThreadLocalStorage

IThreadLocalStorage is a mechanism for creating per-thread data storage in a typesafe and convenient way. Most operating systems provide a mechanism to have data that is replicated for each thread. This provides a very easy means for subsystems to maintain information or context for multiple threads that might be concurrently using its services. However, the operating system interfaces for such storage is usually clunky and typeless. IThreadLocalStorage brings these APIs to you in a much more powerful way.

When you create a thread local storage object, it automatically sets up per-thread storage for you. It lazily creates storage as required so it is efficient. Since it is a template class, you must specialize it for a particular class type, which makes it typesafe as well. Once an object is created, each thread can treat it as though it were the only user of the object.

In effect, IThreadLocalStorage is like a smart pointer except that it provides a unique pointer for each thread.

Class header file: iperthr.hpp

IThreadLocalStorage - Hierarchy List

IThreadLocalBase

IThreadLocalStorage

IThreadLocalStorage - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IThreadLocalStorage objects.

~IThreadLocalStorage

```
public:~IThreadLocalStorage()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor will free the data object for the thread in which the destructor occurs. The data object for other threads that have used this object are not freed. They will be freed when those threads call freeAllStorage().

IThreadLocalStorage

The thread local storage constructors set up the internal data structures required to track all of the threads that use the new thread local sorage object. It does NOT create any actual data yet. Each thread must create its own data, though it is usually done on the thread's behalf by subsystem code into which the threads call.

Overload 1

```
public:IThreadLocalStorage(const bool resettable)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor does the same as the default except that it allows you to indicate whether a thread's data can be overwritten. If you pass false, then multiple sets of a thread's pointer will cause an IInvalidRequest exception.

Overload 2

```
public:IThreadLocalStorage()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor is the most often used constructor. A default constructed object does not allow multiple sets of the data objects. Therefore, once a thread calls adoptStorage() it cannot call adoptStorage() again to overwrite the previous object. Since many uses of thread local data have this type of semantics, the class provides this checking to ensure that overwrites do not happen accidentally.

Access Operators

These operators allow you to access the stored data pointer. These only allow the calling thread to access the copy of the object that belongs to it.

operator *

```
public:AType& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the data object, when the thread local storage object is dereferenced. This is a convenience to allow you to assign objects to the data object easily or to use the data object directly from the thread local object.

operator ->

```
public:AType* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the data object, when the thread local storage object is dereferenced. This is a convenience to allow you to use the data object directly from the thread local object.

Storage Management Methods

These member functions allow you to manage the object stored in the thread local storage object. These members only work on the copy of the object that belongs to the calling thread.

adoptStorage

```
public:void adoptStorage(AType* toadopt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is called to store the storage for the calling thread. When a thread local object is created, it has no way of knowing how many threads are out there or whether they will ever need to use this object. So the construction of the thread local object does not create the data storage for each thread. That must be done within the context of each thread. If storage() returns 0, there is no object for the calling thread and adoptStorage() should be called to create a new object for the calling thread.

If a thread never accesses a thread local storage object, then no data will ever be created for that thread. This is an efficient and simple approach, but it does mean that you must always check for the existence of the data (and allocate it if not) before accessing it.

Note that each thread local storage object can be set up to allow or disallow multiple sets of the data object. If the object allows it, then calling adoptStorage() again will cause the previous object to be deleted and the new object stored. If the object does not allow it, then an InvalidRequest exception is thrown.

Exception

Exception	Description
IInvalidRequest	is thrown if the calling thread has already stored an object, and this object does not allow resetting of the data value. The ability to reset is indicated during construction.

freeStorage

```
public:void freeStorage()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Frees the data object for the calling thread and will set the pointer for that thread back to 0. Subsequent calls to the same thread local storage object on that thread will return zero, until adoptStorage() is called again to store another data object.

get

```
public:AType* get() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A deprecated method that does the same thing as storage(). It is only maintained for backwards compability.

storage

```
public:AType* storage() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the data object for the calling thread. If adoptStorage() has not been called (or it has but freeStorage() has been called since then), the return will be zero. Otherwise, it will be the pointer to the stored data object for the calling thread. This function does the same thing as the -> operator. The -> operator provides cleaner syntax for accessing the methods of the data object in general.

IThreadLocalStorage - Inherited Member Functions and Data

Inherited Public Functions

IThreadLocalBase			
Definition	**Page Number**	**Definition**	**Page Number**
freeAllStorage	157		

Inherited Public Data

None

Inherited Protected Functions

IThreadLocalBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IThreadLocalBase	156	expandThreadSlots	157
freeStorage	158	getNextSlot	157
getSlot	157	IThreadLocalBase	156
operator =	156	setSlot	157

Inherited Protected Data

None

IThreadMemberFn

The IThreadMemberFn template class is derived from IThreadFn for dispatching C++ member functions to an object on a new thread. The template argument is the class of the object where the dispatched member functions are called.

The constructor for such objects requires the following:
- A reference to the object where the dispatched member function is called
- A pointer to the member function to call.

The member functions must return void and accept no parameters.

Example

```
class MyClass {
public:
    void job()  {
        // Code to run on separate thread.
    }
};

MyClass *myObj;
myObj = new MyClass;
INonGUIThread thread;
thread.start( new IThreadMemberFn<MyClass>( *myObj, MyClass::job ) );
```

You can add support for other type of member functions by deriving a class from IThreadFn.

Example 2

```
class Foo {
public:
    void job(int a, double b, Foo* other) {
        // Code to run on separate thread.
    }
};
class IFooFn : public IThreadFn
{
public:
     typedef void (Foo::* func_t)(int, double, Foo*);
     FooFn(Foo& f, func_t fn, int a, double b, Foo* o)
        : foo(f), func(fn), _a(a), _b(b), _o(o) {}
     virtual void run() {
         (foo.*func)(_a, _b, _o);
     }
private:
     Foo& foo;
     func_t func;
     int _a;
     double _b;
     Foo* _o;
}

Foo *foo = new Foo();
INonGUIThread thread;
thread.start( new IFooFn(*foo, &Foo::job, 1, 1.0, foo)  );
```

Class header file: ingthrd.hpp

IThreadMemberFn - Hierarchy List

IMRefCounted

IThreadFn

IThreadMemberFn - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this template class.

~IThreadMemberFn

```
public:virtual ~IThreadMemberFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the thread member function object.

IThreadMemberFn

```
public:IThreadMemberFn(T& obj, void ( T::* mem ) ( ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You construct objects of this class by specifying an argument of the template argument class and a pointer to a member function of the template argument class.

obj Object of the template argument class **T**.

mem Pointer to a member function of the template argument class. The member function should return a **void** parameter and accept no formal parameters.

Run Function

Use these member functions to run the member function that was specified when an object of this class was created.

run

```
public:virtual void run()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the member function that you specify when you create an object of this class.

IThreadMemberFn - Inherited Member Functions and Data

Inherited Public Functions

IThreadFn			
Definition	**Page Number**	**Definition**	**Page Number**
~IThreadFn run	150 150	IThreadFn	150

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

ITimer

The ITimer class creates and references periodic time-interval-based operations.

Timer identifiers are limited to the value 0x7FFF in all environments to allow portability to systems where this limit is imposed by the operating system.

AIX Considerations

At the end of each interval, a notification request is registered with the system for the next interval. Because of processing delays that may occur in the operating system and applications between the time an interval expires and the next one is scheduled, some drift may occur in the timer.

OS/2 Considerations

ITimer objects use timer identifier values above 1000. Limit your use of timer identifiers to values below 1000 when directly creating or managing timers using operating system calls.

The ITimer class uses the operating system message queue for timer expiration. This can cause timer actions to be delayed if the application is currently tying up the queue. If a timer event is delayed, the next timer event still occurs at the expected interval (there should be no timer drift).

Class header file: itimer.hpp

ITimer - Hierarchy List

ITimer

ITimer - Member Functions and Data by Group

Constructors & Destructor

You can construct, copy, and destruct objects of this class.

Note: The ITimer class does not support user-specified timer identifier values or timers that were not created with the ITimer class.

~ITimer

```
public:virtual ~ITimer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor does not stop the timer or deallocate resources. If you want to stop the timer and deallocate resources, call the function ITimer::stop to halt the started timer.

ITimer

Constructs and ITimer object.

Overload 1

```
public:ITimer(unsigned long timerIdentifier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimer object from the timer ID of an existing ITimer object.

timerIdentifier
A value that represents the timer identifier (ID).

Exception

Exception	Description
IInvalidParameter	The timer identifier value is not valid.

Overload 2

```
public:ITimer( const ICountedPointerTo < ITimerFn >& timerFunction,
        unsigned long timerInterval = 1000 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimer object with a specification of code to be run and a time interval (in thousandths of a second).

timerFunction
A reference to an object of the ICountedPointerTo template class that was created using a class derived from ITimerFn. You must dynamically allocate the ITimerFn object using operator new.

timerInterval
A value that represents the time interval. One second (1000 milliseconds) is the default.

This constructor is equivalent to using the default constructor and calling ITimer::start. Therefore, use this form of the constructor to construct a new ITimer and immediately start it.

Overload 3

```
public:ITimer(const ITimer& timer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimer object from an existing ITimer object.

timer A reference to an existing timer object.

Overload 4

```
public:ITimer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimer object using the default constructor. When calling ITimer::start to subsequently start the timer, you can specify a timer interval and a function for the timer to call each time it expires.

Exception

Exception	Description
IOutOfSystemResource	The system is out of available timer identifiers.

Assignment

Use these members to assign one ITimer object to another.

operator =

```
public:ITimer& operator =(const ITimer& timer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns an ITimer object to reference an existing timer.

Comparison Operators

Use these member functions to compare ITimer objects.

operator ==

```
public:bool operator ==(const ITimer timer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if and only if the timers are identical.

Diagnostics

Use these member functions for diagnostic purposes. They return IString representations of the timer diagnostics.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a representation of the timer as diagnostic information.

asString

```
public:virtual IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a string of the form ″ITimer(ID)″, where **ID** is replaced by the actual identifier of the timer.

Expiration Interval

After you start a timer, it expires each time a specified time interval passes. Use these member functions to access the timer expiration interval.

interval

```
public:unsigned long interval() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the timer interval (in thousandths of a second).

setInterval

```
public:virtual ITimer& setInterval(unsigned long interval = 1000)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the timer interval (in thousandths of a second). If no interval is set for a timer, then an interval of one second is used by default.

If this member function is called for a started timer, then the timer is stopped and restarted with the new interval. The ITimerFn being used is saved and used when the timer is restarted.

An interval of 0 causes the timer to expire as quickly as possible.

OS/2 Considerations

When you run a timer on the OS/2 operating system, an effective minimum of about 1/18 of a second exists for timer intervals.

Retrieving Objects

Use these member functions to retrieve a timer object using the class ITimer::Cursor.

timerAt

```
public:static ITimer timerAt(const Cursor& cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ITimer object at the position indicated by **cursor**.

Exception

Exception	Description
IInvalidParameter	The ITimer::Cursor is invalid.

Starting and Stopping

Use these member functions to start and stop timers.

isStarted

```
public:bool isStarted() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the timer is currently active. A timer is not active if you have stopped it or have never started it. The timer will continue to expire as long as it is active.

start

Starts a time-interval-based operation with an arbitrary ITimerFn object. The ITimerFn is passed in using an ICountedPointerTo object so that ITimer can reference count the ITimerFn object and subsequently delete it when you stop the timer.

ITimer calls the ITimerFn object each time the timer interval expires, until you stop the timer.

Overload 1

```
public:virtual ITimer&
  start( const ICountedPointerTo < ITimerFn >& timerFunction )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

timerFunction

A reference to an object of the ICountedPointerTo template class that was created using a class derived from ITimerFn. You must dynamically allocate the ITimerFn object using operator new.

Exception

Exception	Description
IInvalidRequest	The timer has already been started.
IAccessError	The operating system is unable to start the timer.

Overload 2

```
public:virtual ITimer&
  start( const ICountedPointerTo < ITimerFn >& timerFunction,
         unsigned long timerInterval )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

timerFunction

A reference to an object of the ICountedPointerTo template class that was created using a class derived from ITimerFn. You must dynamically allocate the ITimerFn object using operator new.

timerInterval

A value that represents the timer interval.

stop

```
public:virtual ITimer& stop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the timer. You must stop the timer to deallocate resources used by the timer.

Exception

Exception	Description
IAccessError	The operating system is unable to stop the timer.

Timer Information

Use these member functions to access timer information, such as the timer identifier.

id

```
public:unsigned long id() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the timer's identifier.

ITimer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITimer::Cursor

The nested class ITimer::Cursor iterates over the active timers for the current application. This class iterates the timers running on any of the application's threads, although it only considers timers started with the ITimer class.

Class header file: itimer.hpp

ITimer::Cursor - Hierarchy List

ITimer::Cursor

ITimer::Cursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~Cursor

```
public:virtual ~Cursor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

Cursor

```
public:Cursor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You can only construct objects of this class with the default constructor, which does not require any arguments.

Timer Iteration

Use these member functions to iterate through the set of active timers.

invalidate

```
public:virtual void invalidate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Invalidates the cursor.

isValid

```
public:virtual bool isValid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor is valid.

setToFirst

```
public:virtual bool setToFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes the cursor to the first timer. If no timers are active, then it returns false and invalidates the cursor.

setToNext

```
public:virtual bool setToNext()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the cursor to the next timer. If no other timers are active, then it returns false and invalidates the cursor.

If you call setToNext with an invalid cursor, it sets the cursor to the first active timer.

ITimer::Cursor - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITimerFn

The ITimerFn class is an abstract timer function class that you use with the class ITimer. To use ITimerFn, create a class derived from ITimerFn that implements the ITimerFn::timerExpired function. ITimerMemberFn and ITimerMemberFn0 are template classes that derive from ITimerFn.

The ITimerFn class represents functions that an active timer (represented by ITimer) calls when it expires. ITimer reference-counts ITimerFn objects to manage their destruction when you stop the timer. To assign an ITimerFn object to an ITimer, create the ITimerFn object using operator new and pass its address to the ITimer object either when you construct it or when you call ITimer::start.

Class header file: itimer.hpp

ITimerFn - Hierarchy List

IMRefCounted

ITimerFn

ITimerMemberFn

ITimerMemberFn0

ITimerFn - Member Functions and Data by Group

Constructors & Destructor

You can destruct objects of this class, although ITimer generally manages the destruction of these objects for you. You cannot construct objects of this class because ITimerFn is an abstract class. You cannot copy or assign ITimerFn objects because both the copy constructor and the assignment operator are private members.

~ITimerFn

```
public:virtual ~ITimerFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITimerFn object.

ITimerFn

```
public:ITimerFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITimerFn provides only a default constructor, which does not require any arguments.

If you are using this class with the ITimer class, use operator new to create objects of classes derived from ITimerFn, because ITimer reference counts these objects.

Timer Expiration

After you start a timer, it expires each time a specified time interval passes. Use these members to perform actions each time the timer expires.

timerExpired

```
public:virtual void timerExpired(unsigned long timerId) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Function to be called by an ITimer object each time the timer expires.

ITimerFn - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

ITimerMemberFn

The ITimerMemberFn template class is an ITimerFn-derived class for dispatching C++ member functions to an object when a timer expires. The template argument is as follows:

T The class of the object against which the member functions are applied.

Use this class instead of ITimerMemberFn0 if you need the timer identifier (ID).

Objects of this class call C++ member functions when a timer expires. The template argument is the class of the object for which the functions are called. The constructor for such objects require a reference to the object that the member functions are to be applied to and a pointer to the member functions.

The member functions return void and accept a timer identifier (ID) argument.

Class header file: itimer.hpp

ITimerMemberFn - Hierarchy List

IMRefCounted

ITimerFn

ITimerMemberFn

ITimerMemberFn - Member Functions and Data by Group

Constructors & Destructor

You can only construct and destruct objects of this class, although ITimer generally manages the destruction of these objects for you. You cannot copy or assign ITimerMemberFn objects because both the copy constructor and the assignment operator are private members.

~ITimerMemberFn

```
public:virtual ~ITimerMemberFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITimerMemberFn

```
public:ITimerMemberFn( T& object,
                       void ( T::* memberFunction ) ( unsigned long ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs objects of this class.

Use operator new to create objects of this class if you will be using them with ITimer, because ITimer reference counts these objects.

object A reference to an object of the template argument class **T**.

memberFunction

A pointer to a member function of the template argument class. The member function should return a **void** parameter and accept an **unsigned long** formal parameter that represents the timer ID.

Timer Expiration

After you start a timer, it expires each time a specified time interval passes. Use these members to perform actions each time the timer expires.

timerExpired

```
public:virtual void timerExpired(unsigned long timerId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the pointed-to member function that was provided at construction each time the timer expires.

ITimerMemberFn - Inherited Member Functions and Data

Inherited Public Functions

ITimerFn			
Definition	**Page Number**	**Definition**	**Page Number**
~ITimerFn	170	ITimerFn	170
timerExpired	171		

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef	57	count	58
removeRef	57		

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted	57	IMRefCounted	57
operator =	57		

Inherited Protected Data

None

ITimerMemberFn0

The ITimerMemberFn0 template class is an ITimerFn-derived class that calls a C++ member function when a timer expires. The template argument is as follows:

T The class of the object against which the member functions are applied.

Use this class instead of ITimerMemberFn if you do not need the timer identifier (ID).

Objects of this class dispatch C++ member functions when a timer expires. The template argument is the class of the object for which the functions are called. The constructor for such objects require a reference to the object that the member functions are to be applied to and a pointer to the member functions.

The member functions return void and accept no arguments.

Class header file: itimer.hpp

ITimerMemberFn0 - Hierarchy List

IMRefCounted

ITimerFn

ITimerMemberFn0

ITimerMemberFn0 - Member Functions and Data by Group

Constructors & Destructor

You can only construct and destruct objects of this class, although ITimer generally manages the destruction of these objects for you. You cannot copy or assign ITimerMemberFn0 objects because both the copy constructor and the assignment operator are private members.

~ITimerMemberFn0

```
public:virtual ~ITimerMemberFn0()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITimerMemberFn0

```
public:ITimerMemberFn0( T& object,
                        void ( T::* memberFunction ) ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs objects of this class.

Use operator new to create objects of this class if you will be using them with ITimer, because ITimer reference counts these objects.

obj A reference to an object of the template argument class **T**.

memberFunction

A pointer to a member function of the template argument class. The member function should return a **void** parameter and accept no formal parameters.

Timer Expiration

After you start a timer, it expires each time a specified time interval passes. Use these members to perform actions each time the timer expires.

timerExpired

```
public:virtual void timerExpired(unsigned long timerId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the pointed-to member function that was provided at construction each time the timer expires.

ITimerMemberFn0 - Inherited Member Functions and Data

Inherited Public Functions

ITimerFn			
Definition	**Page Number**	**Definition**	**Page Number**
~ITimerFn timerExpired	170 171	ITimerFn	170

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

IVersion

IVersion represents a specific version number. It provides mechanisms for version comparison and for querying the Open Class Library version. An IVersion is composed of:

- A major number
- A minor number

To query the current Open Class Library version, use the static function libraryVersion. For example:

```
IVersion IOCVersion(IVersion::libraryVersion());
cout << IOCVersion.asString() << endl;
```

Class header file: iversion.hpp

IVersion - Hierarchy List

IVersion

IVersion - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destructs objects of this class.

IVersion

Constructs an IVersion object.

Overload 1

```
public:IVersion(const IVersion& aVersion)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a version object by copying another version object.

aVersion

The source version object to copy construct from.

Overload 2

```
public:IVersion(unsigned short aMajor, unsigned short aMinor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to construct a version object from the given parameters.

aMajor

The major version number for the new object.

aMinor

The minor version number for the new object.

Overload 3

```
public:IVersion()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a default IVersion object.

Comparison Operators

Use these operators to compare IVersion objects.

operator !=

```
public:bool operator !=(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for inequality.

aVersion

The other version object to compare to this one for inequality.

Return

true if this object is not equal to the passed object, else false.

operator <

```
public:bool operator <(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for relative magnitude.

aVersion
The version to compare to this object for relative magnitude.
Return
true if this object represents a lower version than the passed object, else false.

operator <=

```
public:bool operator <=(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for relative magnitude.
aVersion
The version to compare to this object for relative magnitude.
Return
true if this object represents a lower (or equivalent) version than the passed object, else false.

operator ==

```
public:bool operator ==(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for equality.
aVersion
The other version object to compare to this one for equality.
Return
true if this object represents an equivalent version as the passed object, else false.

operator >

```
public:bool operator >(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for relative magnitude.
aVersion
The version to compare to this object for relative magnitude.
Return
true if this object represents a greater version than the passed object, else false.

operator >=

```
public:bool operator >=(const IVersion& aVersion) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function compares this object to another version object for relative magnitude.
aVersion
The version to compare to this object for relative magnitude.
Return
true if this object represents a greater (or equivalent) version than the passed object, else false.

Diagnostics

Use these member functions for diagnostic purposes. They return an IString representation of an object of this class.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the version object as an IString. By default, the string is formatted as Major.Minor.

Member Access

Use these member functions to get and set the major and minor version numbers.

major

```
public:unsigned short major()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the major version number of the version object.

minor

```
public:unsigned short minor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the minor version number of the version object.

setMajor

```
public:void setMajor(unsigned short aMajor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function sets the major version number of the version object.

aMajor

The new major version number.

setMinor

```
public:void setMinor(unsigned short aMinor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function sets the minor version number of the version object.

aMinor

The new minor version number.

Open Class Library Version

Use this member function to obtain the IBM Open Class Library version.

libraryVersion

```
public:static IVersion libraryVersion()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns a version object which contains the IBM Open Class Library version.

Streaming

Use these member functions to stream in and out IVersion objects.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function streams this object in from the passed data stream. Any existing content is overwritten by the streamed in information.

fromWhere

The stream to stream this object in from.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function streams this IVersion object out to the passed data stream.

toWhere

The stream to stream this object out to.

readFromStream

```
protected:void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method supports the streaming of a version object from a data stream.

writeToStream

```
protected:void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method supports the streaming of a version object to a data stream.

IVersion - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWaitAbandoned

This exception is thrown when the host abandons a wait before timing out.

Class header file: ithrdexc.hpp

IWaitAbandoned - Hierarchy List

IException

IWaitAbandoned

IWaitAbandoned - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IWaitAbandoned object from another.

~IWaitAbandoned

```
public:virtual ~IWaitAbandoned()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the exception object.

IWaitAbandoned

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IWaitAbandoned(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Overload 2

```
public:IWaitAbandoned(const IWaitAbandoned& toCopy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IWaitAbandoned - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769

IException			
Definition	**Page Number**	**Definition**	**Page Number**
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

std

Class header file: iccsid.hpp

std - Hierarchy List

std

std - Member Functions and Data by Group

std - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 2. Streams

IAddressAlreadyInContext

IAddressAlreadyInContext is an exception class thrown by the streaming classes.

This exception may be thrown while reading from or writing C++ objects to an IDataStream. It indicates that there is an internal inconsistency with the datastream having to do with the handling of aliased objects, and could be caused by corruption of the stream data, or by programming errors in the application or by defects in the streaming classes.

Class header file: istrmexc.hpp

IAddressAlreadyInContext - Hierarchy List

IException

IAddressAlreadyInContext

IAddressAlreadyInContext - Member Functions and Data by Group

Constructors & Destructor

Use the destructor to destroy an instance of an IAddressAlreadyInContext object. Instances are only constructed by the streaming implementation code, never by application code.

~IAddressAlreadyInContext

```
public:virtual ~IAddressAlreadyInContext()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IAddressAlreadyInContext

Constructs an IAddressAlreadyInContext object.

Overload 1

```
public:IAddressAlreadyInContext( const char* text = 0,
                                 ioc::numeric_t errorID = 0,
                                 Severity severity = recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IAddressAlreadyInContext( const IAddressAlreadyInContext& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

© Copyright IBM Corp. 2001

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IAddressAlreadyInContext - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IDataStream

IDataStream provides a facility for C++ classes to read or write their state to or from an external place - a file, memory, or other destination. Similar facilities in other systems are called internalization and externalization, or flatten and resurrection. An IDataStream object represents the place from or to which objects may be read or written.

This class does not support multi-threaded applications.

Do not derive from this class.

The primary functions for reading and writing objects from an IDataStream are the following global function:

```
readObject()
writeObject()
readAliasedObject()
writeAliasedObject()
```

Other related classes are IMStreamable, IStreamModule, IStreamContextFrame, IStreamInFrame, and IStreamOutFrame.

Class header file: idatstrm.hpp

IDataStream - Hierarchy List

IDataStream

IDataStream - Member Functions and Data by Group

Constructors & Destructor

The constructors for this class are all private to prevent construction of these objects on the stack. Use the **Factory methods** to create instances of this class on the heap.

~IDataStream

```
public:~IDataStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

Factory methods

Use these member functions to create IDataStream objects. Also see IFile::createStream() for the creation of an IDataStream that targets files.

createMemoryStream

```
public:static IDataStream*
  createMemoryStream( int encoding = kInteroperableBinary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an IDataStream that writes and reads data to a temporary memory buffer. The buffer is automatically allocated and grows as required to hold the data.

This form of IDataStream is commonly used to test the streaming operators for other (user) classes.

encoding
The type of stream encoding to use. Defaults to kInteroperableBinary.

Read and Write Array

These member functions read and write array of primitive types.

readBools

```
public:void readBools(bool* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of bool values from the IDataStream.
data Address of the array into which the data will be read.
howMany
Number of bool values to be read.

readBytes

```
public:void readBytes(char* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of bytes from the data stream.
data Address of the buffer into which the data will be placed.
howMany
Number of bytes to read from the stream.

readDoubles

```
public:void readDoubles(double* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of double values from the IDataStream.
data Address of the array into which the data will be read.
howMany
Number of double values to be read.

readFloats

```
public:void readFloats(float* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of floating point values from an IDataStream.
data Address of the array into which the data will be read.
howMany
Number of floating point values to be read from the stream.

readInts

```
public:void readInts(int* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of int values from an IDataStream.
data Address of the array into which the data will be read.
howMany
Number of int values to be read.

readLongLongs

```
public:void
  readLongLongs( long long* data,
                 ioc::unumeric_t howMany )
```

This is supported on **AIX OS/400 z/OS**

Reads an array of long long values from an IDataStream.
data Address of the array into which the data will be read.
howMany
Number of long long values to be read.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

readLongs

```
public:void readLongs(long* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of long values from an IDataStream.
data Address of the array into which the data will be read.
howMany
Number of long values to be read.

readShorts

```
public:void readShorts(short* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an array of short values from an IDataStream.
data Address of the array into which the data will be read
howMany
Number of short values to be read from the stream.

writeBools

```
public:void writeBools(const bool* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of bool values to an IDataStream.
data Address of the buffer containing the data to be written.
howMany
Number of bool values to be written to the stream.

writeBytes

```
public:void writeBytes(const char* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of bytes to an IDataStream.
data Address of the buffer containing the data to be written to the stream.
howMany
Number of bytes to write to the stream.

writeDoubles

```
public:void
  writeDoubles( const double* data,
                ioc::unumeric_t howMany )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of double values to an IDataStream.
data Address of the buffer containing the data to be written.

howMany
Number of double values to be written to the stream.

writeFloats

```
public:void
  writeFloats( const float* data,
               ioc::unumeric_t howMany )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of floating point values to an IDataStream.
data Address of the buffer containing the data to be written to the stream.
howMany
Number of floating point values to be written.

writeInts

```
public:void writeInts(const int* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of int values to an IDataStream.
data Address of the buffer holding the data to be written to the stream.
howMany
Number of int values to be written.

writeLongLongs

```
public:void
  writeLongLongs( const long long* data,
                  ioc::unumeric_t howMany )
```

This is supported on **AIX OS/400 z/OS**

Writes an array of long long values to an IDataStream.
data Address of the buffer holding the data to be written to the stream.
howMany
Number of long long values to be written.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

writeLongs

```
public:void writeLongs(const long* data, ioc::unumeric_t howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of long values to an IDataStream.
data Address of the buffer holding the data to be written to the stream.
howMany
Number of long values to be written.

writeShorts

```
public:void
  writeShorts( const short* data,
               ioc::unumeric_t howMany )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an array of short values to an IDataStream.
data Address of the buffer containing the data to write to the IDataStream.

howMany
Number of short values to be written.

Read and Write Primitive Types

These member functions read and write primitive types.

readBool

```
public:bool readBool()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads a bool value from the stream. Also see the global stream-in function <<=, which will perform the same operation, but is more convenient to use.

Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

readByte

```
public:char readByte()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads one byte of data from the stream. Also see the global stream-in function <<=, which will perform the same operation, but is more convenient to use.

Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

readDouble

```
public:double readDouble()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads a double value from the IDataStream.

Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

readFloat

```
public:float readFloat()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads a floating point value from an IDataStream. See also the global operator <<=(float &, IDataSteam &), which provides a more convenient way to read data from an IDataStream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

readInt

```
public:int readInt()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads an int value from an IDataStream. The global operator <<=(long &, IDataStream &) provides a more convenient way to perform this operation.

readLong

```
public:long readLong()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads a long value from an IDataStream. The global operator <<=(long &, IDataStream &) provides a more convenient way to perform this operation.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

readLongLong

```
public:long long readLongLong()
```

This is supported on **AIX OS/400 z/OS**

Reads a long long value from an IDataStream. The global operator <<=(long long &, IDataStream &) provides a more convenient way to perform this operation.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

readShort

```
public:short readShort()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads a short value from an IDataStream. The global operator <<=(short &, IDataStream &) provides a more convenient way to perform this operation.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.

Exception	Description
IEndOfStream	Operation reached the end of the stream.

writeBool

```
public:void writeBool(bool source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a bool value to an IDataStream. The global operator >>= (bool, IDataStream &) provides a more convenient way to perform this operation.
source Value to be written to the stream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

writeByte

```
public:void writeByte(char source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a byte to an IDataStream. The global operator >>=(char, IDataStream) provides a more convenient way to perform this operation.
source Data to be written.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

writeDouble

```
public:void writeDouble(double source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a double precision floating point value to an IDataStream. The global operator >>= (double, IDataStream &) provides a more convenient way to perform this operation.
source Value to be written to the stream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

writeFloat

```
public:void writeFloat(float source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a floating point value to an IDataStream. The global streaming operator >>=(float, IDataStream &) provides a more convenient way of performing this operation.
source Value to be written to the stream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

writeInt

```
public:void writeInt(int source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes an int value to an IDataStream. The global streaming operator >>= (long, IDataStream &) provides a more convenient way to perform this operation.
source Value to be written to the stream.

writeLong

```
public:void writeLong(long source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a long value to an IDataStream. The global streaming operator >>= (long, IDataStream &) provides a more convenient way to perform this operation.
source Value to be written to the stream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

writeLongLong

```
public:void writeLongLong(long long source)
```

This is supported on **AIX OS/400 z/OS**

Writes a long long value to an IDataStream. The global streaming operator >>= (long long, IDataStream &) provides a more convenient way to perform this operation.
source Value to be written to the stream.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

writeShort

```
public:void writeShort(short source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a short value to an IDataStream. The global streaming operator >>=(short, IDataStream) provides a more convenient way of performing this operation.
source Value to be written to the stream.
Exception

Exception	Description
IInvalidDataOnStream	Data found on the stream was not of the correct type.
IEndOfStream	Operation reached the end of the stream.

Stream Control

These member functions affect the state of the the stream rather than the data which is in it.
flush

```
public:void flush()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Forces all data out through the stream buffer.
reset

```
public:void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the stream back to the beginning.

IDataStream - Enumerations

EStreamEncoding

Defines encoding types.

IDataStream - Inherited Member Functions and Data

Inherited Public Functions

IDataStream			
Definition	**Page Number**	**Definition**	**Page Number**
`streamIO`	185		

Inherited Public Data

None
Inherited Protected Functions

None
Inherited Protected Data

None

IEndOfStream

IEndOfStream is an exception class thrown by the streaming classes.

This exception signals a read past the end of an IDataStream.

Class header file: istrmexc.hpp

IEndOfStream - Hierarchy List

IException

IInvalidRequest

IEndOfStream

IEndOfStream - Member Functions and Data by Group

Constructors & Destructor

Use the destructor to destroy an instance of an IEndOfStream object. Instances are only constructed by the streaming implementation code, never by application code.

~IEndOfStream

```
public:virtual ~IEndOfStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IEndOfStream

Constructs an IEndOfStream object.

Overload 1

```
public:IEndOfStream( const char* text = 0,
                     ioc::numeric_t errorID = 0,
                     Severity severity = recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IEndOfStream(const IEndOfStream& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IEndOfStream - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IInvalidRequest			
Definition	**Page Number**	**Definition**	**Page Number**
~IInvalidRequest	781	IInvalidRequest	781
nameW	782		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IInvalidContextNumber

IInvalidContextNumber is an exception class thrown by the streaming classes.

This exception signals that an object alias read from an IDataStream refers to non-existent data. This is an internal consistency check; a failure generally indicates that the stream data has been corrupted, or that some other serious progamming error has occured.

Class header file: istrmexc.hpp

IInvalidContextNumber - Hierarchy List

IException

IInvalidContextNumber

IInvalidContextNumber - Member Functions and Data by Group

Constructors & Destructor

Use the destructor to destroy an instance of an IInvalidContextNumber object. Instances are only constructed by the streaming implementation code, never by application code.

~IInvalidContextNumber

```
public:virtual ~IInvalidContextNumber()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IInvalidContextNumber

Constructs an IInvalidContextNumber object.

Overload 1

```
public:IInvalidContextNumber( const char* text = 0,
                              ioc::numeric_t errorID = 0,
                              Severity severity = recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IInvalidContextNumber(const IInvalidContextNumber& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IInvalidContextNumber - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IInvalidDataOnStream

IInvalidDataOnStream is an exception class thrown by the streaming classes.

This exception signals invalid data read from an IDataStream.

Class header file: istrmexc.hpp

IInvalidDataOnStream - Hierarchy List

IException

IInvalidDataOnStream

IInvalidDataOnStream - Member Functions and Data by Group

Constructors & Destructor

Use the destructor to destroy an instance of an IInvalidDataOnStream object. Instances are only constructed by the streaming implementation code, never by application code.

~IInvalidDataOnStream

```
public:virtual ~IInvalidDataOnStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IInvalidDataOnStream

Constructs an IInvalidDataOnStream object.

Overload 1

```
public:IInvalidDataOnStream( const char* text = 0,
                             ioc::numeric_t errorID = 0,
                             Severity severity = recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IInvalidDataOnStream(const IInvalidDataOnStream& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IInvalidDataOnStream - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769

IException			
Definition	**Page Number**	**Definition**	**Page Number**
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IMStreamable

IMStreamable is a common base class for streamable classes.

For further information on streaming, see the descriptions of classes IDataStream, IStreamModule, IStreamContextFrame, IStreamInFrame and IStreamOutFrame.

IMStreamable is required for streamable classes that support **polymorphic streaming** and/or **release-to-release data compatibility**.

Polymorphic streaming means that the application defined objects will be used with the ::readObject() and ::writeObject() functions, that the object to be read or written will be specified by a reference, and that the actual object type may be any derived type of the specified reference type.

Release-to-release data compatibility is the ability for a class to interpret stream data that was written either by older or newer versions of the class. For more details on RRDC, see classes IStreamInFrame and IStreamOutFrame.

Simple classes (or structs, or primitives) that do not need these advanced capabilities may be streamed even though they do not derive from IMStreamable.

IMStreamable is an abstract class. All non-abstract classes deriving from IMStreamable must incude the StreamableDeclarations and StreamableDefinitions macros.

Example:

```
//
//  Class decaration, typically in a header file.
//
class MyApplicationClass: public virtual IMStreamable {
public:
   ...
   StreamableDeclarationsMacro(MyApplicationClass);

protected:
   virtual void writeToStream(IDataStream &toWhere) const;
   virtual void readFromStream(IDataStream &fromWhere);

private:
   ...
};

//
//  These go in the  implementation (.cpp) file for the class.
//
IStreamModule  myStreamModule("a_name_unique_to_my_app");
StreamableDefinitionsMacro(MyApplicationClass, myStreamModule);
 . . .
```

Do not derive your own classes from this class.

Class header file: imstrmbl.hpp

IMStreamable - Hierarchy List

IMStreamable

IFileSystemEntity

ICompoundName

ILocaleKey

IMStreamable - Member Functions and Data by Group

Constructors & Destructor

Constructor and destructor for IMStreamable objects.

~IMStreamable

```
public:virtual ~IMStreamable()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IMStreamable

Constructs an object of this class.

Overload 1

```
protected:IMStreamable(const IMStreamable& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor will be implicitly called by the copy constructor for classes deriving from IMStreamable. Stand-alone

instances of IMStreamable cannot exist, and thus cannot be copied; IMStreamable can only exist as a base class of other derived classes.

Overload 2

```
protected:IMStreamable()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor will be implicitly called by classes deriving from IMStreamable. Stand-alone instances of IMStreamable can not be constructed; they can only exist as a base class of other derived classes.

Implementation Functions

These member functions are used by the streaming implementation classes. User code should not override them or call them directly.

externalizeToStream

```
public:void externalizeToStream(IDataStream& towhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is used by the global streaming opertors and IStreamIO implementation classes. User code should not override this function or call it directly.

internalizeFromStream

```
public:void internalizeFromStream(IDataStream& fromwhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is used by the global streaming opertors and IStreamIO implementation classes. User code should not override this function or call it directly.

typeRepresentation

```
public:virtual const ITypeRepresentation&
  typeRepresentation() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is declared and implemented by the Streamable macros, and is part of the implementation of the streaming classes. User code should not override this function or call it directly.

Streaming Interface

These member functions used to write and read data to and from streams.

readFromStream

```
protected:virtual void readFromStream(IDataStream& fromwhere) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

All subclasses of IMStreamable must override this function.

The override reads the state of this object - instance variables and base classes - from the IDataStream. Instance variables are read using the <<= operator or ::readObject() functions; base classes are read by calling the inherited readFromStream function.

The streaming implementation will call this function in response to user code invoking either the <<= operator or a ::readObject function on an instance of the class.

The writeToStream() function must write the data to the stream in the same order in which it will be read by this function.

writeToStream

```
protected:virtual void writeToStream(IDataStream& towhere) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

All subclasses of IMStreamable must override this function.

The override writes the state of the object - instance variables and base classes - to the IDataStream. Instance variables are written using the >>= operator or ::writeObject() functions; base classes are written by calling the inherited writeToStream function.

The streaming implementation will call this function in response to user code invoking either the >>= operator or a ::writeObject function on an instance of the class.

The readFromStream() function must read the data from the stream in the same order in which it was written by this function.

IMStreamable - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamContextFrame

This class is used when streaming a group of related aliased objects.

Aliased streaming comes into play when dealing with complex objects (such as collections) that may have several references (pointers) to some other object. When streaming out the complex object, we only want to write the ″other″ object once, rather than repeating it for each pointer in the main object. And, when streaming the complex object back in, we want each of the pointers that originally referred to the single ″other″ object to once again refer to a single instance of ″other″, not **n** repeated instances.

To accomplish this, the streaming implementation establishes a context and remembers all objects that are streamed during the duration of that context. A context will automatically be established whenever a read or write aliased object is performed and persist through any recursive calls to read/write aliased objects that are required to complete the original operation.

Class IStreamContextFrame allows a stream context to be explicitly established so that it can persist through the streaming of a group of related objects. The construction of the IStreamContextFrame establishes the context; the subsequent destruction terminates the stream contex.

Explicit (user created) IStreamContexts are not required when a context is needed only for the duration of the streaming of a single complex object.

IStreamContextFrame objects are intended to be created as local (stack) variables, which allows the context to be automatically shut down when the local variable goes out of scope at the end of the block in which it is declared.

This class is not thread aware.

Do not derive from this class.

Example of use:

```
void write3RelatedObjects(IDataStream &toWhere, IMRefCounted &obj1,
IMRefCounted &obj2, IMRefCounted &obj3)
{
    IStreamContextFrame aContext(toWhere);
    writeAliasedObject(obj1, toWhere);
    writeAliasedObject(obj2, toWhere);
    writeAliasedObject(obj3, toWhere);
}
// The context is ended when the IStreamContextFrame object goes out of scope.
```

Class header file: istrmcf.hpp

IStreamContextFrame - Hierarchy List

IStreamContextFrame

IStreamContextFrame - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IStreamContextFrame

```
public:~IStreamContextFrame()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor ends the stream context. Since IStreamContextFrames are always allocated as local variables, the destructor will be implicitly invoked when the block containing the declaration goes out of scope. Explicit calls to the destructor are not used.

IStreamContextFrame

```
public:IStreamContextFrame(IDataStream& dataStream)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IStreamContextFrame object.

dataStream
The data stream which this context object will track.

IStreamContextFrame - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamInFrame

This class is used for **release-to-release data compatibility (RRDC)**.

RRDC allows a new version of a class to read in instance data streamed by an earlier version. It also allows an older class implementation to read data written by a newer implementation.

The streaming system is described in more detail in the description for classes IMStreamable and IDataStream.

For any application class that supports RRDC, the override of IMStreamable::readFromStream() must declare a local instance of IStreamInFrame before streaming in the member data of the class. The scope of this local variable must include all of the stream-ins of the member data.

In addition, readFromStream() function for any class that is in a second or higher release, and has extended the data it streams, must make a check at the point that it has read all of the data that would have been written by the older releases. If the stream was written by an older version, and thus does not contain the additional data, the readFromStream function must set the state of the object to a reasonable value and not attempt to stream in the missing data. IStreamInFrame::atEnd() allows this check to be made.

Use of stream frames is required in classes, even in their first release, if RRDC may be required for subsequent releases.

Example:

```
//
//  An example of a class and readFromStream function for the first
//  release of the class.  RRDC with future releases is allowed for,
//  through the use of IStreamInFrame.
//
class myClass: public IMStreamable {
    ...
    int             fData1;
    someType   fData2;
    ...
};

void myClass::readFromStream(IDataStream &fromWhere) {
    IStreamInFrame frame(fromWhere);
    fData1 <<= fromWhere;
    fData2 <<= fromWhere;
};
    // When the IStreamInFrame goes out of scope, as it does here,
```

```
// any extra data that may have been written by some future, unknown
// to us at this time, version of myClass will be automatically skipped,
// leaving the stream position correct for the next object to be streamed in.
```

Class header file: imstrmbl.hpp

IStreamInFrame - Hierarchy List

IStreamInFrame

IStreamInFrame - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IStreamInFrame objects.

~IStreamInFrame

```
public:~IStreamInFrame()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor. IStreamInFrame objects are always allocated on the stack, so the destructor is invoked automatically when the variable goes out of scope.

IStreamInFrame

```
public:IStreamInFrame(IDataStream& aStream)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a new instance of an IStreamInFrame object. This will always be a local (stack) variable within the readFromStream() method of a streamable class.

Stream Info

Use these member functions to query information about the stream.

atEnd

```
public:bool atEnd()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks whether all of the data that was written by a class's writeToStream() function within the context of an IStreamOutFrame has been read within the context of this IStreamInFrame.

This function (and class) is only used within the implementation of a readFromStream() function of a streamable class.

The readFromStream() function for any class that is in a second, or higher, release and has extended the data that it writes to a stream should call atEnd() at the point(s) where all of the data written by earlier stream formats has been read. A return value of true indicates that the stream data was indeed written by an earlier version, and that there is no more data available to be read by this stream-in function. When this occurs, the readFromStream function should default all data items that have not yet been read, without reading any additional values from the stream.

Return

true if at end; false otherwise.

IStreamInFrame - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamIO

IStreamIO is an abstract class that forms part of the implementation for an IDataStream. IStreamIO defines the protocol by which IDataStreams actually read and write primitive types; subclasses of IStreamIO define the actual format of the data as it appears on the stream.

Each of the three stream encodings supported by the streaming classes (Interoperable Binary, Raw Binary, and Debug) is implemented by means of a specific concrete subclass of IStreamIO.

Class header file: istrmio.hpp

IStreamIO - Hierarchy List

IStreamIO

IStreamIO - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IStreamIO

```
protected:virtual ~IStreamIO()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStreamIO

Constructs an IStreamIO object.

Overload 1

```
protected:IStreamIO(const IStreamIO& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a copy of the given stream where **other** is the stream to copy.

Overload 2

```
protected:IStreamIO()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates and initializes a new stream.

Establishing and Clearing Context

Use these member functions to establish and clear a stream context.

clearContext

```
protected:virtual void clearContext() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Clears a stream context.

establishContext

```
protected:virtual bool establishContext() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Establishes a context for streaming and returns a bool.

Manipulating Streams

Use these member functions to set, reset, and flush a data stream.

flush

```
protected:virtual void flush() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Forces all data out through the stream buffer.

reset

```
protected:virtual void reset() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the stream back to the beginning.

setStream

```
protected:virtual void setStream(IDataStream* dataStream) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the data stream. This function is called from the IDataStream constructor so that the IStreamIO can stream subobjects.

Streaming Arrays

Use these member functions to stream arrays of primitive types.

readBools

```
protected:virtual void
  readBools( bool* data,
             ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of bool data.

data The first element of the array.

howMany
The number of elements in the array.

readBytes

```
protected:virtual void
  readBytes( char* data,
             ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of bytes of data.

data The first element of the array.

howMany
The number of elements in the array.

readDoubles

```
protected:virtual void
  readDoubles( double* data,
               ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of double-precision floating point values.

data The first element of the array.

howMany
The number of elements in the array.

readFloats

```
protected:virtual void
  readFloats( float* data,
              ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of floating point values.

data The first element of the array.

howMany
The number of elements in the array.

readInts

```
protected:virtual void
  readInts( int* data,
            ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

readLongLongs

```
protected:virtual void
  readLongLongs( long long* data,
                 ioc::unumeric_t howMany ) = 0
```

This is supported on **AIX OS/400 z/OS**

Streams in an array of long long integers.

data The first element of the array.

howMany
The number of elements in the array.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

readLongs

```
protected:virtual void
  readLongs( long* data,
             ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of long integers.

data The first element of the array.

howMany
The number of elements in the array.

readShorts

```
protected:virtual void
  readShorts( short* data,
              ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an array of short integers.

data The first element of the array.

howMany
The number of elements in the array.

writeBools

```
protected:virtual void
  writeBools( const bool* data,
              ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array of bool data.

data The first element of the array.

howMany
The number of elements in the array.

writeBytes

```
protected:virtual void
  writeBytes( const char* data,
              ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array containing bytes of data.

data The first element of the array.

howMany
The number of elements in the array.

writeDoubles

```
protected:virtual void
  writeDoubles( const double* data,
                ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array of double-precision floating point values.

data The first element of the array.

howMany
The number of elements in the array.

writeFloats

```
protected:virtual void
  writeFloats( const float* data,
               ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array of floating point values.

data The first element of the array.

howMany
The number of elements in the array.

writeInts

```
protected:virtual void
  writeInts( const int* data,
             ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

writeLongLongs

```
protected:virtual void
  writeLongLongs( const long long* data,
                  ioc::unumeric_t howMany ) = 0
```

This is supported on **AIX OS/400 z/OS**

Streams out the specified array of long long integers.

data The first element of the array.

howMany
The number of elements in the array.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

writeLongs

```
protected:virtual void
  writeLongs( const long* data,
              ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array of long integers.

data The first element of the array.

howMany
The number of elements in the array.

writeShorts

```
protected:virtual void
  writeShorts( const short* data,
               ioc::unumeric_t howMany ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified array of short integers.

data The first element of the array.

howMany
The number of elements in the array.

Streaming Frames

Use these member functions to stream and detect the beginning and end of a stream frame.

atEndOfFrame

```
protected:virtual bool atEndOfFrame() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Detects the end of a stream frame. Returns true if the end of the stream frame is found.

readBeginFrame

```
protected:virtual void readBeginFrame() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in the beginning of a stream frame.

readEndFrame

```
protected:virtual void readEndFrame() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in the end of a stream frame.

writeBeginFrame

```
protected:virtual void writeBeginFrame() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the beginning of a stream frame.

writeEndFrame

```
protected:virtual void writeEndFrame() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the end of a stream frame.

Streaming Objects

Use these member functions to stream streamable objects.

readAliasedObject

```
protected:virtual bool
  readAliasedObject( IMStreamable *& object,
                     IMRefCounted *& refCounted,
                     const IStreamFinder* finder,
                     const ITypeRepresentation* typeRep ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an aliased, streamable object and returns a bool.

object On return, a pointer to the object.

refCounted
On return, a pointer to the reference counted object.

finder A pointer to the stream finder.

typeRep
A pointer to the type representation of the object.

readObject

```
protected:virtual IMStreamable*
  readObject( const IStreamFinder* finder,
              const ITypeRepresentation* typeRep ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a streamable object.

finder A pointer to the stream finder.

typeRep
A pointer to the type representation of the object.

writeAliasedObject

```
protected:virtual bool
  writeAliasedObject( const IMStreamable* object,
                      const IMRefCounted* refCounted,
                      const ITypeRepresentation* typeRep ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the aliased, streamable object and returns a bool.

object A pointer to the object to stream.

refCounted
A pointer to the reference counted object.

typeRep
A pointer to the type representation of the object.

writeObject

```
protected:virtual void
  writeObject( const IMStreamable* object,
               const ITypeRepresentation* typeRep ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the streamable object.

object A pointer to the object to stream out.

typeRep
A pointer to the type representation of the object.

Streaming Primitive Types

Use these member functions to stream primitive types.

readBool

```
protected:virtual bool readBool() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a bool data.

readByte

```
protected:virtual char readByte() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a byte of data.

readDouble

```
protected:virtual double readDouble() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a double-precision floating point value.

readFloat

```
protected:virtual float readFloat() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a floating point value.

readInt

```
protected:virtual int readInt() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in an integer.

readLong

```
protected:virtual long readLong() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a long integer.

readLongLong

```
protected:virtual long long readLongLong() = 0
```

This is supported on **AIX OS/400 z/OS**

Streams in a long long integer.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

readShort

```
protected:virtual short readShort() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in a short integer.

writeBool

```
protected:virtual void writeBool(bool source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified bool data.

source The value to stream out.

writeByte

```
protected:virtual void writeByte(char source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified byte of data.

source The value to stream out.

writeDouble

```
protected:virtual void writeDouble(double source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified double-precision floating point value.

source The value to stream out.

writeFloat

```
protected:virtual void writeFloat(float source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified floating point value.

source The value to stream out.

writeInt

```
protected:virtual void writeInt(int source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified integer.

source The value to stream out.

writeLong

```
protected:virtual void writeLong(long source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified long integer.

source The value to stream out.

writeLongLong

```
protected:virtual void writeLongLong(long long source) = 0
```

This is supported on **AIX OS/400 z/OS**

Streams out the specified long long integer.

source The value to stream out.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

writeShort

```
protected:virtual void writeShort(short source) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams out the specified short integer.

source The value to stream out.

Supporting Subclasses

These member functions are used to support subclasses.

createUninitialized

```
protected:static IMStreamable*
  createUninitialized( const IStreamFinder* finder,
                       const ITypeName& typeName,
                       IMRefCounted *& refCounted )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is for the internal use of the Open Class Library. It should not be used by your own derived classes.

findTypeRepresentation

```
protected:static const ITypeRepresentation*
  findTypeRepresentation( const IStreamFinder* finder,
                          const ITypeName& typeName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is for the internal use of the Open Class Library. It should not be used by your own derived classes.

IStreamIO - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamModule

IStreamModules are used with streaming (IDataStreams). They provide a name-space like facility to help avoid conflicts between the names of classes that may be written to a stream.

Class name ambiguity becomes an issue with polymorphic streaming (::writeObject() and ::readObject()). WriteToStream() puts a representation of the class type on the stream in addition to the class data, and readFromStream picks up the class name and instantiates an object of that type which then receives data from the stream.

For example:

```
// include header imstrmbl.hpp which has StreamableDeclarationsMacro and
//    StreamableDefinitionsMacro
#include  < imstrmbl.hpp >
//
//  Declare a streamable class
//
class MyClass: public virtual IMStreamable {
   // add in declarations for streaming
   StreamableDeclarationsMacro(MyClass);
   ...
};

//
//  Implementation for the class
//

//  The stream module into which MyClass will be scoped.
IStreamModule gMyStreamModule("my_product_name_and_my_company_name");

//  Associate MyClass with this stream module
StreamableDefinitionsMacro(MyClass, gMyStreamModule);
```

```
//  Implementation of the member functions of MyClass...
MyClass::MyClass() {
  ...
}
```

Do not derive from this class.

This class is not thread aware.

Class header file: istrmmod.hpp

IStreamModule - Hierarchy List

IStreamModule

IStreamModule - Member Functions and Data by Group

Constructors & Destructor

The constructor and destructor for this class.

~IStreamModule

```
public:virtual ~IStreamModule()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStreamModule

```
public:IStreamModule(const char* moduleName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a named IStreamModule. Instances are always static (or global) variables.

moduleName
The name that is used, together with the C++ class name, to uniquely identify types on a stream.

Choose the name such that other developers will not pick the same one.

IStreamModule - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamOutFrame

This class is used for **release-to-release data compatibility (RRDC)**.

RRDC allows a user class to stream out its instance data in such a way that either newer or older versions of the class will be able to succsessfully read the data.

For any application class that supports RRDC, the override of IMStreamable::writeToStream() must declare a local instance of IStreamOutFrame before streaming out the member data of the class. The scope of this local variable must enclose all of the stream-outs of the member data.

The effect of the the presence of an IStreamOutFrame on the IDataStream is to delimit, or frame, the data that is written to the stream. This allows the reader of the stream to determine whether the stream has too little data (old writer, new reader), the right amount of data (compatible reader and writer), or too much data (new writer, old reader). In the case of too much data, the frame provides the information needed to automatically skip over the extra data.

Use of stream frames is required in classes, even in their first release, if RRDC may be required for subsequent releases.

Example:

```
//
//  An example of a class and writeToStream function.
//  RRDC with other versions of the class is allowed
//  for through the use of IStreamOutFrame.
//
class myClass: public IMStreamable {
    ...
    int              fData1;
    someType   fData2;
    ...
};

void myClass:writeToStream(IDataStream &toWhere) {
    IStreamOutFrame frame(toWhere);
    fData1 >>= toWhere;
    fData2 >>= toWhere;
};
```

Class header file: imstrmbl.hpp

IStreamOutFrame - Hierarchy List

IStreamOutFrame

IStreamOutFrame - Member Functions and Data by Group

Constructors & Destructor

The constructor and destructor for this class.

~IStreamOutFrame

```
public:~IStreamOutFrame()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStreamOutFrame objects are always allocated on the stack, so the destructor is invoked automatically when the variable goes out of scope.

IStreamOutFrame

```
public:IStreamOutFrame(IDataStream& aStream)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IStreamOutFrame object. This will normally be a local variable within the writeToStream() method of a streamable class.

aStream

The stream that is being written.

IStreamOutFrame - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IUnknownTypeOnStream

IUnknownTypeOnStream is an exception class thrown by the streaming framework.

This exception signals an object with an unknown type was encountered while reading from an IDataStream. One situation that can cause this error is that the dynamic link library, or shared library, with the class implementation for the type is not loaded in the process that is reading from the stream.

Class header file: istrmexc.hpp

IUnknownTypeOnStream - Hierarchy List

IException

IUnknownTypeOnStream

IUnknownTypeOnStream - Member Functions and Data by Group

Constructors & Destructor

Use the destructor to destroy an instance of an IUnknownTypeOnStream object. Instances are only constructed by the streaming implementation code, never by application code.

~IUnknownTypeOnStream

```
public:virtual ~IUnknownTypeOnStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IUnknownTypeOnStream

Constructs an IUnknownTypeOnStream object.

Overload 1

```
public:IUnknownTypeOnStream( const char* text = 0,
                             ioc::numeric_t errorID = 0,
                             Severity severity = recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IUnknownTypeOnStream(const IUnknownTypeOnStream& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IUnknownTypeOnStream - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 3. File Systems

ICompoundName

ICompoundName represents paths for files, network resources, and other named objects.

Class header file: icmpdnam.hpp

ICompoundName - Hierarchy List

IMStreamable

ICompoundName

IPathName

ICompoundName - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct compound name objects.

~ICompoundName

Destructor.

```
public:virtual ~ICompoundName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICompoundName object.

ICompoundName

Constructs a ICompoundName object.

Overload 1

```
public:ICompoundName( const IText& path,
                      const ICompoundNameParser& parser )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function uses the specified parser to construct a compound name from the specified text.

path The text from which to construct the compound name.

parser The parser to use to construct the compound name from the text.

Overload 2

```
public:ICompoundName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Overload 3

```
public:ICompoundName(const ICompoundName& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

© Copyright IBM Corp. 2001

Use this member function to construct a copy of the given object.

copyMe

The object to copy.

operator =

Assignment operator.

```
public:ICompoundName& operator =(const ICompoundName& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one ICompoundName object to another.

copyMe

The object to copy.

Accessing Path Components

These member functions allow you to access individual path components.

componentAt

```
public:virtual IText componentAt(ioc::unumeric_t index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the component at the given index of the path. The first component is at index = 1.

index The index of the component to return.

lastComponent

```
public:virtual IText lastComponent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the last component of the path.

numberOfComponents

```
public:virtual ioc::unumeric_t numberOfComponents() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of components in the path.

Comparing and Querying

Use these member functions to compare compound names and query information about them.

hash

```
public:virtual ioc::numeric_t hash() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the hash value.

isFullPath

```
public:virtual bool isFullPath() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines if this is a full path. This function returns true if the object is a full path, false if it is a relative path.

operator !=

```
public:virtual bool
  operator !=( const ICompoundName& anotherPath ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this inequality operator to test for inequality with the given object.

anotherPath
The object to test for inequality.

Returns a bool value that indicates whether the objects are equal or not.

operator ==

```
public:virtual bool
  operator ==( const ICompoundName& anotherPath ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this equality operator to test for equality with the given object.

anotherPath
The object to test for equality.

Returns a bool value that indicates whether the objects are equal or not.

setFullPath

Make full path.

```
public:virtual void setFullPath(bool isFull)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes this path a full path name.

isFull The value indicating whether to make the object a full path (true) or not (false).

Manipulating Paths

These member functions allow you to add and remove name segments of a compound name.

append

```
public:virtual void append(const ICompoundName& relativePath)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Append one path to another.

relativePath
The path to append.

Exception

Exception	Description
IInvalidParameter	Thrown if relativePath is a full path.

appendComponent

```
public:virtual void appendComponent(const IText& componentPath)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Append a single component to the path.

componentPath
The component to append.

factorCommonRoot

```
public:virtual void
  factorCommonRoot( const ICompoundName& anotherPath,
                    ICompoundName& commonRoot,
                    ICompoundName& rootToLeaf ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Remove the path name components common to two path names.

If two paths have common roots, as in /usr/include/sys/mode.h (in "this") and /usr/include/Foo.h (in "anotherPath"), this function finds the common part (in "commonRoot"), leaving (in "rootToLeaf"), sys/mode.h.

anotherPath
The path to compare against.

commonRoot
On output, specifies the components common to both paths.

rootToLeaf
On output, specifies the components of this path that were not common to the other path.

truncate

Remove components from the end of this path.

Overload 1

```
public:virtual void truncate(ioc::unumeric_t segmentsToRemove)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void truncate(const ICompoundName& relativePath)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the components of another path from the end of this path.

Truncate is similar to factorCommonRoot, but works from the "end" of the path, not the beginning. If "this" path is /usr/include/sys/mode.h, and "relativePath" is sys/mode.h, this member function will change "this" to be /usr/include.

relativePath
The path to remove from the end of this path.

Exception

Exception	Description
IInvalidParameter	Thrown if relativePath does not exist at the end of this path.

Streaming

These member functions support the streaming of compound name objects.

readFromStream

```
public:virtual void readFromStream(IDataStream& fromwhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads the object from the given data stream.

fromwhere
From where to stream the data.

writeToStream

```
public:virtual void writeToStream(IDataStream& towhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the object to the given data stream.

towhere
Where to stream the data.

ICompoundName - Inherited Member Functions and Data

Inherited Public Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable	200	readFromStream	201
writeToStream	202		

Inherited Protected Data

None

ICompoundNameParser

ICompoundNameParser is an abstract base class that converts between ICompoundName and IText.

Use one of the concrete subclasses when you need a parser.

Do not derive your own classes from this class.

Class header file: icmpdnam.hpp

ICompoundNameParser - Hierarchy List

ICompoundNameParser
- IOS400PathNameParser
- IUnixPathNameParser
- IWin32PathNameParser

ICompoundNameParser - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct compound name parser objects.

~ICompoundNameParser

```
public:virtual ~ICompoundNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICompoundNameParser object.

ICompoundNameParser

Constructs a new ICompoundName object.

Overload 1

```
protected:ICompoundNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor.

Overload 2

```
protected:ICompoundNameParser(const ICompoundNameParser&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this copy constructor to construct a copy of the given object.

operator =

```
protected:ICompoundNameParser& operator =(const ICompoundNameParser&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one ICompoundNameParser object to another.

Accessing the Separator

Use this member function to access the separator character for the target file system.

separator

```
public:virtual ioc::unichar_t separator() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Unicode character that is used to separate path name components.

Comparison Operators

Use these operators to compare compound name parser objects.

operator !=

```
public:bool operator !=(const ICompoundNameParser&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this inequality operator to test for inequality with the given object.

Return

A bool value that indicates whether the objects were equal or not.

operator ==

```
public:virtual bool operator ==(const ICompoundNameParser&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this equality operator to test for equality with the given object.

Return

A bool value that indicates whether the objects where equal or not.

Identifying the Type

Use these member functions to obtain information about the type of parser.

typeOf

```
protected:virtual IParserType typeOf() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a string that uniquely identifies this parser's C++ class.

Typically, this function simply returns the class name. When compiler RTTI support is available on all platforms, this member function will be removed.

Parsing and Formatting

Use these member functions to parse and format compound names according to the rules of the derived class' target file system.

formatName

```
public:virtual IText formatName(const ICompoundName&) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Convert an ICompoundName into a platform-specific text string. Returns an IText object with the formatted name.

parseName

```
public:virtual ICompoundName parseName(const IText&) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Convert a platform-specific text string into an ICompoundName object. Returns the compound name object with the parsed text.

ICompoundNameParser - Type Definitions

IParserType

```
typedef const char * IParserType
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IParserType is used to uniquely identify a parser's C++ class.

ICompoundNameParser - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IDirectory

This class provides a surrogate object for a file system directory.

Do not derive your own classes from this class.

Class header file: ifilesys.hpp

IDirectory - Hierarchy List

IMStreamable

IFileSystemEntity

IDirectory

IVolume

IDirectory - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct IDirectory objects.

~IDirectory

```
public:virtual ~IDirectory()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the IDirectory object.

IDirectory

Constructs an IDirectory object.

Overload 1

```
public:IDirectory(const IDirectory& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the copy constructor to create a directory object copied from the given one.

Overload 2

```
public:IDirectory()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an empty directory object.

Assignment Operator

Use this assignment operator to copy the given object to another IDirectory object.

operator =

```
public:IDirectory& operator =(const IFileSystemEntity& right)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The assignment operator for this class.
right The object to copy.

Browsing and Managing the File System

Use these member functions to browse and manage the file system, for example, to create a new directory or file, delete the contents of a directory, create a temporary directory, and so forth.

createDirectory

```
public:IDirectory
  createDirectory( const IFileName& pathOrName,
                   const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a new directory in the specified location relative to this directory.

pathOrName
The host-specific path name of the directory to be created. This can be a single ″leaf″ name of a directory to be created in this directory, or it can be a partial path relative to this directory.

ICompoundNameParser
The parser used with the name of the new directory. If none is specified, the system's default parser is used.

Exception

Exception	Description
IInvalidParameter	Thrown if pathOrName is a full path instead of a partial path.

createFile

Creates a new file in the specified location relative to this directory.

Overload 1

```
public:IFile
  createFile( const IFileName& name,
              IFileSize initialSize = 0,
              const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a new file in the specified location relative to this directory.

name The host-specific path name of the file to be created. This can be a single ″leaf″ name of a file to be created in this directory, or it can be a partial path relative to this directory.

initialSize
The initial size of the created file, in bytes. If none is specified, the file is empty.

ICompoundNameParser
The parser used with the name of the new file. If none is specified, the system's default parser is used.

Overload 2

```
public:IFile
  createFile( const ICCSId& id,
              const IFileName& name,
              IFileSize initialSize = 0,
              const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

Creates a new file in the specified location relative to this directory.

id The code page, or character set id, with which to open the file. If the value is 0, the code page or coded character set id of the application will be used.

name The host-specific path name of the file to be created. This can be a single "leaf" name of a file to be created in this directory, or it can be a partial path relative to this directory.

initialSize
The initial size of the created file, in bytes. If none is specified, the file is empty.

ICompoundNameParser
The parser used with the name of the new file. If none is specified, the system's default parser is used.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

createTemporary

```
public:static IDirectory
  createTemporary( const IFileName& dirName = uniqueName ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a temporary directory in the appropriate place for temporary storage on the current system. The directory is temporary in that it is not automatically deleted when the IDirectory object is destroyed.

dirName
The name for the temporary directory. If none is provided, a unique name is generated.

Exception

Exception	Description
IEntityExists	Thrown if a name is provided and a temporary directory with that name already exists.

deleteAllContents

```
public:void deleteAllContents()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes all entities within this directory. This includes the contents of any subdirectories.

getRecord

```
public:IFSDirectoryRecord* getRecord() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file system record.

lookUp

Finds the entity specified by the parameters.

Overload 1

```
public:IFileSystemEntity
  lookUp( const IPathName& partialPath ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to find by a portable path relative to this directory.

partialPath

The portable path name of the entity to be looked up. This can be a single component path containing the ″leaf″ name of an entity in this directory, or it can be a partial path relative to this directory.

Exception

Exception	Description
IInvalidParameter	Thrown if partialPath is a full path instead of a partial path.

Overload 2

```
public:IFileSystemEntity
  lookUp( const IHostPathName& pathOrName,
          const ICompoundNameParser& = defaultParser ( ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to find the entity specified by a host-specific path relative to this directory.

pathOrName

The host-specific path name of the entity to be looked up. This can be a single ″leaf″ name of an entity in this directory, or it can be a partial path relative to this directory.

ICompoundNameParser

The parser used with the name of the new directory. If none is specified, the system's default parser is used.

Exception

Exception	Description
IInvalidParameter	Thrown if pathOrName is a full path instead of a partial path.

Class Constants

Constants for use with this class.

kKind

```
public:static const IFileKind kKind
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The kind of the file system entity. A homegrown type of RTTI for the filesystem.

Getting and Setting the Processes Current Directory

Use these member functions to set or get the current directory for the process.

current

```
public:static IDirectory current()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current directory for this process. File system operations using partial path names are evaluated relative to the current directory.

setCurrent

```
public:static void setCurrent(const IDirectory&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the current directory for this process. Subsequent file system operations using partial path names are evaluated relative to the current directory.

Special Downcast Construction

Use these constructors to create IDirectory objects based on the given information, such as a host path name, a partial path name, a file system entity, or a file system directory record.

IDirectory

Constructs an IDirectory object.

Overload 1

```
public:IDirectory(IFSDirectoryRecord* Record)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a directory object from the given file system directory record.

Overload 2

```
public:IDirectory( const IHostPathName& path,
            ECreateFlags flags = kMustExist,
            const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a directory object from a host-specific path name.

path Text string containing the host-specific path name.

flags Specifies whether the constructor creates or overwrites a directory on the disk.

parser The path name parser which should be used to parse the provided path. If no parser is specified, the default for the current system is used.

Overload 3

```
public:IDirectory(const IFileSystemEntity& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this copy constructor to create a directory object from the given file system entity.

copy If this entity refers to a physical directory, the constructed IDirectory refers to the same object. If it is not a directory, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a directory.

Overload 4

```
public:IDirectory( const IPathName& path,
            ECreateFlags flags = kMustExist )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a directory object from the given portable path name.

path The portable path name.

flags Specifies whether the constructor creates or overwrites a directory on the disk.

IDirectory - Inherited Member Functions and Data

Inherited Public Functions

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileSystemEntity	281	connected	286
copyTo	284	defaultParser	283
deleteSelf	284	getRecord	282
hostPath	282	IFileSystemEntity	281
isA	283	modificationTime	282
moveTo	284	name	282
operator !=	286	operator =	282
operator ==	286	parent	285
path	283	readFromStream	285
setName	283	temporary	283
topmostParent	285	uniqueName	284
valid	286	volume	285
writeToStream	285		

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable writeToStream	200 202	readFromStream	201

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
setInvalid	286	typeSafeAssign	283

Inherited Protected Data

None

IDirectoryIterator

IDirectoryIterator is used to iterate over the contents of a directory. It can be used to iterate a single directory or to recursively iterate a directory and all of its subdirectories.

This class is designed to mimic a C++ pointer, similar to the iterator classes in the ANSI/ISO C++ draft standard. When the C++ Standard Library is available, IDirectoryIterator will inherit from input_iterator<IFileSystemEntity>.

Typical usage is as follows:

```
IDirectory myDirectory = ....;

for (IDirectoryIterator anEntity(myDirectory); anEntity; anEntity++)
{
    DoSomething( *anEntity );
}
```

Class header file: ifileitr.hpp

IDirectoryIterator - Hierarchy List

IDirectoryIterator

IDirectoryIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct directory iterators.

~IDirectoryIterator

```
public:virtual ~IDirectoryIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IDirectoryIterator

Constructs an invalid, past-the-end iterator that you can use for iterating over the contents of a given directory.

Overload 1

```
public:IDirectoryIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Overload 2

```
public:IDirectoryIterator(const IDirectoryIterator& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor.

Copying a directory iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

Overload 3

```
public:IDirectoryIterator( const IDirectory&,
                    bool recursive = false )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an iterator over the contents of a given directory

directory
The directory over which to iterate

recursive
If true, also iterates over the contents of subdirectories. The default is false, to iterate over a single directory.

operator =

```
public:IDirectoryIterator&
  operator =( const IDirectoryIterator& copy )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IDirectoryIterator object to another.

Assigning a directory iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

Comparison and validation operators

Use these operators to compare iterators and to confirm the validity of iterators.

operator !=

```
public:bool operator !=(const IDirectoryIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this inequality operator to test the inequality of two directory iterators.

Two directory iterators compare as equal if they are both past the end of their iteration, or if one was copy-constructed from the other.

operator ==

```
public:bool operator ==(const IDirectoryIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this equality operator to test the equality of two directory iterators.

Two directory iterators compare as equal if they are both past the end of their iteration, or if one was copy-constructed from the other.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to test the validity of the iterator.

This member function is often used in the termination condition of a for or while loop. Returns false when there are no more entities over which to iterate.

Member Access

Use these member functions to access data of this class.

recursive

```
public:bool recursive() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this iteration is recursive.

Standard iteration functions

Use these iteration operators to iterate through the directories.

operator *

```
public:IFileSystemEntity operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current entity in the iteration. This member function allows the iterator to be used as if it were a normal C++ pointer, in expressions like:

```
FunctionCall( *anIterator )
```

operator ++

Pre-increment operator used for expressions like ++iterator. Proceeds to the next entity in the iteration.

Overload 1

```
public:IDirectoryIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IDirectoryIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ->

```
public:IFileSystemEntity* operator ->()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the current entity in the iteration. Allows the iterator to be used as if it were a normal C++ pointer, in expressions like:

```
anIterator->deleteSelf()
```

Miscellaneous Members

end

```
public:static const IDirectoryIterator& end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an iterator which represents the end of iteration.

IDirectoryIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEntityInUse

IEntityInUse is an exception that indicates that an attempt has been made to move, delete, or operate on an entity which is currently in use.

Class header file: iexcfile.hpp

IEntityInUse - Hierarchy List

IException

IAccessError

IEntityInUse

IEntityInUse - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct objects of this class. You cannot assign one IEntityInUse object from another.

~IEntityInUse

```
public:virtual ~IEntityInUse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IEntityInUse

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.

- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IEntityInUse(const IEntityInUse&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IEntityInUse(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function constructs an exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IEntityInUse - Inherited Member Functions and Data

Inherited Public Functions

IAccessError			
Definition	**Page Number**	**Definition**	**Page Number**
~IAccessError	731	IAccessError	731
nameW	732		

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769

IException			
Definition	**Page Number**	**Definition**	**Page Number**
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IEntityInvalid

IEntityInvalid is an exception that indicates that an operation was attempted on an invalid IFileSystemEntity. This exception is thrown when there is an attempt to perform an operation on an IFileSystemEntity object which does not point to a valid on-disk entity.

Class header file: iexcfile.hpp

IEntityInvalid - Hierarchy List

IException

IInvalidParameter

IEntityInvalid

IEntityInvalid - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IEntityInvalid object from another.

~IEntityInvalid

```
public:virtual ~IEntityInvalid()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IEntityInvalid

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IEntityInvalid(const IEntityInvalid&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IEntityInvalid(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function constructs an exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IEntityInvalid - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768

IException			
Definition	**Page Number**	**Definition**	**Page Number**
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IInvalidParameter			
Definition	**Page Number**	**Definition**	**Page Number**
~IInvalidParameter	778	IInvalidParameter	778
nameW	779		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IEntityTypeMismatch

IEntityTypeMismatch is an exception that indicates an attempted assignment to the wrong type of entity object.

Class header file: iexcfile.hpp

IEntityTypeMismatch - Hierarchy List

IException

IInvalidParameter

IEntityTypeMismatch

IEntityTypeMismatch - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IEntityTypeMismatch object from another.

~IEntityTypeMismatch

```
public:virtual ~IEntityTypeMismatch()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IEntityTypeMismatch

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the constructor where you specify the type of the expected file entity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

Copy constructor.

```
public:IEntityTypeMismatch(const IEntityTypeMismatch&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IEntityTypeMismatch( IFileKind kind,
                    const IFileSystemEntity& entity )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function constructs an exception with a localized message specific to the given entity type.

entity The entity whose invalid type triggered the exception.
kind The type of entity that was expected.

Overload 3

```
public:IEntityTypeMismatch(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function constructs an exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IEntityTypeMismatch - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IInvalidParameter			
Definition	**Page Number**	**Definition**	**Page Number**
~IInvalidParameter	778	IInvalidParameter	778
nameW	779		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IFifoFile

This class provides a surrogate object for a FIFO file.

Do not derive your own classes from this class.

Note: A FIFO file must be created before using this class. To create a FIFO file, use mkfifo(path, mode) where path is the pathname of the FIFO file being created, and mode is the mode with which the FIFO file is to be opened.

Class header file: ifilesys.hpp

IFifoFile - Hierarchy List

IMStreamable

IFileSystemEntity

IFifoFile

IFifoFile - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct objects of this class.

~IFifoFile

```
public:virtual ~IFifoFile()
```

This is supported on **AIX OS/400 z/OS**

Destroys and object of this class.

IFifoFile

Constructs an IFifoFile object.

This constructor can be used only if a FIFO file already exists.

Overload 1

```
public:IFifoFile()
```

This is supported on **AIX OS/400 z/OS**

The default constructor.

Overload 2

```
public:IFifoFile(const IFifoFile& copy)
```

This is supported on **AIX OS/400 z/OS**

Use the copy constructor to create an IFifoFile object copied from the given one.

Assignment Operator

Use this operator to replace the existing object with the one specified.

operator =

```
public:IFifoFile& operator =(const IFileSystemEntity& right)
```

This is supported on **AIX OS/400 z/OS**

The assignment operator.

copy If this entity refers to a FIFO file, the IFifoFile now refers to the same object. If not, an exception is thrown.

Class Constants

The kind of the file system entity. A form of homegrown RTTI.

kKind

```
public:static const IFileKind kKind
```

This is supported on **AIX OS/400 z/OS**

An IFileKind constant that identifies the IFifoFile class. This constant is used for dynamic type checking.

Setting and Getting FIFO file Values

Use these member functions to set a FIFO file's modification time and get the size of a FIFO file.

getRecord

```
public:IFSFifoRecord* getRecord() const
```

This is supported on **AIX OS/400 z/OS**

Returns the file system record.

setModificationTime

```
public:void
  setModificationTime( const ITimeStamp& modificationTime ) const
```

This is supported on **AIX OS/400 z/OS**

Sets the FIFO file's modification time. Subsequent calls to IFileSystemEntity::modificationTime return the new value.

size

```
public:IFileSize size() const
```

This is supported on **AIX OS/400 z/OS**

Returns the size of the file, in bytes.

Special Downcast Constructors

Use these constructors to create IFifoFile objects based on the given information, such as a host path name, a partial path name, a file system entity, or a file system directory record.

IFifoFile

Constructs an IFifoFile object.

This constructor can be used only if a FIFO file already exists.

Overload 1

```
public:IFifoFile( const IHostPathName& path,
          ECreateFlags flags = kMustExist,
          const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

Use this constructor to create an IFifoFile object from a host-specific path name.

path Text string containing the host-specific path name.

flags Specifies whether the constructor creates or overwrites a file on the disk.

parser The path name parser which should be used to parse the provided path. If no parser is specified, the default for the current system is used.

Overload 2

```
public:IFifoFile( const IPathName& path,
          ECreateFlags flags = kMustExist )
```

This is supported on **AIX OS/400 z/OS**

Use this constructor to create an IFifoFile object from the given portable path name.

path The portable path name.

flags Specifies whether the constructor creates or overwrites a file on the disk.

Overload 3

```
public:IFifoFile(IFSFifoRecord* Record)
```

This is supported on **AIX OS/400 z/OS**

Use this constructor to create an IFifoFile object from the given file system record.

Overload 4

```
public:IFifoFile(const IFileSystemEntity& copy)
```

This is supported on **AIX OS/400 z/OS**

Use this constructor to create an IFifoFile object that is constructed from a copy of the given file system entity, with type checking.

Streaming Support

Use this member function to create a stream.

createStream

Creates an IBM Open Class Library stream for I/O on this FIFO file.

Overload 1

```
public:IDataStream*
  createStream( const ICCSId& id,
                int modeForMe = in + out,
                int modeForOthers = 0,
                IDataStream::EStreamEncoding encoding =
                IDataStream::kInteroperableBinary )
```

This is supported on **AIX OS/400 z/OS**

Creates an IBM Open Class Library stream for I/O on this FIFO file.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. See IFifoIOStream::IFifoIOStream for details.

modeForOthers
Specifies the modes in which you want to allow others to open the file concurrently. See IFifoIOStream::IFifoIOStream for details.

encoding
The format for data on the stream. See IDataStream::EStreamEncoding for details.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:IDataStream*
  createStream( int modeForMe = in + out,
                int modeForOthers = 0,
                IDataStream::EStreamEncoding encoding =
                IDataStream::kInteroperableBinary )
```

This is supported on **AIX OS/400 z/OS**

Creates an IBM Open Class Library stream for I/O on this FIFO file.

modeForMe
The mode in which the file should be opened. See IFifoIOStream::IFifoIOStream for details.

modeForOthers
Specifies the modes in which you want to allow others to open the file concurrently. See IFifoIOStream::IFifoIOStream for details.

encoding
The format for data on the stream. See IDataStream::EStreamEncoding for details.

IFifoFile - Enumerations

EOpenMode

Enumerator that defines the mode with which a file is to be opened.

Variation 1

```
enum EOpenMode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 binary=0200,
                 text=0400 }
```

This is supported on **AIX OS/400 z/OS**

Variation 2

```
enum EOpenMode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 binary=0200 }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFifoFile - Inherited Member Functions and Data

Inherited Public Functions

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileSystemEntity	281	connected	286
copyTo	284	defaultParser	283
deleteSelf	284	getRecord	282
hostPath	282	IFileSystemEntity	281
isA	283	modificationTime	282
moveTo	284	name	282
operator !=	286	operator =	282
operator ==	286	parent	285
path	283	readFromStream	285
setName	283	temporary	283
topmostParent	285	uniqueName	284
valid	286	volume	285
writeToStream	285		

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable	200	readFromStream	201
writeToStream	202		

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
setInvalid	286	typeSafeAssign	283

Inherited Protected Data

None

IFifoStdIOStream

Standard-library-style stream class which can be constructed from an IFifoFile.

This class allows ANSI iostream-based code to interoperate with the IBM Open Class file system API. IFifoStdIOStream class consists mostly of constructors and open/close member functions. To perform I/O operations, use the member functions inherited from iostream.

NOTE: IFifoIOStream is a typedef of either IFifoUslIOStream or IFifoStdIOStream. When __IOC_ANSI_STREAM is defined, IFifoIOStream is a typedef of IFifoStdIOStream.

Class header file: ififostr.hpp

IFifoStdIOStream - Hierarchy List

IFifoStdIOStream

IFifoStdIOStream - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IFifoStdIOStream

```
public:virtual ~IFifoStdIOStream()
```

This is supported on **AIX OS/400 z/OS**

Destroys an object of this class. If the stream is still open, the destructor attempts to close it. However, it does not throw an exception if the close fails.

IFifoStdIOStream

Constructs an IFileStdIOStream object.

Overload 1

```
public:IFifoStdIOStream( const IHostPathName& path,
                         int modeForMe = std::ios::in + std::ios::out,
                         int modeForOthers = 0,
                         const ICompoundNameParser& = IFifoFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the FIFO file which should be opened.

modeForMe

The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.

- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Overload 2

```
public:IFifoStdIOStream( const IFifoFile& fileToOpen,
                 int modeForMe = std::ios::in + std::ios::out,
                 int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The FIFO file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Overload 3

```
public:IFifoStdIOStream( const IHostPathName& path,
                 const ICCSId& id,
                 int modeForMe = std::ios::in + std::ios::out,
                 int modeForOthers = 0,
                 const ICompoundNameParser& = IFifoFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the FIFO file which should be opened.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 4

```
public:IFifoStdIOStream( const IFifoFile& fileToOpen,
                 const ICCSId& id,
                 int modeForMe = std::ios::in + std::ios::out,
                 int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The FIFO file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.

- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers

Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Member Access

Use these member functions to access the data of this class.

file

```
public:IFifoFile file() const
```

This is supported on **AIX OS/400 z/OS**

Returns the FIFO file to which this stream is attached.

Opening and Closing Streams

Use these member functions to open and close the stream.

close

```
public:void close()
```

This is supported on **AIX OS/400 z/OS**

Closes the stream if it is open. Unlike the destructor, this member function does throw an exception if it fails.

open

Opens the stream.

Overload 1

```
public:void
  open( const IFifoFile& fileToOpen,
        int modeForMe = std::ios::in + std::ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another FIFO file. If the stream is already open, it closes current file first.

fileToOpen

The FIFO file to be opened for streaming.

modeForMe

The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.

- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Overload 2

```
public:void
  open( const IFifoFile& fileToOpen,
        const ICCSId& id,
        int modeForMe = std::ios::in + std::ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another FIFO file. If the stream is already open, it closes current file first.

fileToOpen
The FIFO file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

IFifoStdIOStream - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IFifoUslIOStream

Standard-library-style stream class which can be constructed from an IFifoFile.

This class allows iostream-based code to interoperate with the IBM Open Class file system API. IFifoUslIOStream class consists mostly of constructors and open/close member functions. To perform I/O operations, use the member functions inherited from iostream.

NOTE: IFifoIOStream is a typedef of either IFifoUslIOStream or IFifoStdIOStream.

When the __IOC_ANSI_STREAM is not defined, IFifoIOStream is a typedef of IFifoUslIOStream.

Class header file: ififostr.hpp

IFifoUslIOStream - Hierarchy List

ios

istream

iostream

IFifoUslIOStream

IFifoUslIOStream - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IFifoUslIOStream

```
public:virtual ~IFifoUslIOStream()
```

This is supported on **AIX OS/400 z/OS**

Destroys an object of this class. If the stream is still open, the destructor attempts to close it. However, it does not throw an exception if the close fails.

IFifoUslIOStream

Constructs an IFifoUslIOStream object.

Overload 1

```
public:IFifoUslIOStream( const IHostPathName& path,
                  int modeForMe = ios::in + ios::out,
                  int modeForOthers = 0,
                  const ICompoundNameParser& = IFifoFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the FIFO file which should be opened.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Overload 2

```
public:IFifoUslIOStream( const IFifoFile& fileToOpen,
                  int modeForMe = ios::in + ios::out,
                  int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The FIFO file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.

- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Overload 3

```
public:IFifoUslIOStream( const IHostPathName& path,
                 const ICCSId& id,
                 int modeForMe = ios::in + ios::out,
                 int modeForOthers = 0,
                 const ICompoundNameParser& = IFifoFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the FIFO file which should be opened.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 4

```
public:IFifoUslIOStream( const IFifoFile& fileToOpen,
                         const ICCSId& id,
                         int modeForMe = ios::in + ios::out,
                         int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The FIFO file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.
- ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Member Access

Use these member functions to access the data of this class.

file

```
public:IFifoFile file() const
```

This is supported on **AIX OS/400 z/OS**

Returns the FIFO file to which this stream is attached.

Opening and Closing Streams

Use these member functions to open and close the stream.

close

```
public:void close()
```

This is supported on **AIX OS/400 z/OS**

Closes the stream if it is open. Unlike the destructor, this member function does throw an exception if it fails.

open

Opens the stream.

Overload 1

```
public:void
  open( const IFifoFile& fileToOpen,
        int modeForMe = ios::in + ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another FIFO file. If the stream is already open, it closes current file first.

fileToOpen
The FIFO file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Overload 2

```
public:void
  open( const IFifoFile& fileToOpen,
        const ICCSId& id,
        int modeForMe = ios::in + ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another FIFO file. If the stream is already open, it closes current file first.

fileToOpen
The FIFO file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for

this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers

Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

IFifoUsIIOStream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
~iostream	2955	iostream	2955

istream			
Definition	**Page Number**	**Definition**	**Page Number**
~istream	2961	gcount	2962
get	2962	get_complicated	2966
getline	2966	ignore	2969
ipfx	2984	isfx	2985
istream	2961	operator >>	2972
peek	2970	putback	2983
read	2970	rs_complicated	2971
seekg	2983	sync	2984
tellg	2984		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

istream			
Definition	**Page Number**	**Definition**	**Page Number**
do_ipfx	2985	eatwhite	2972
istream	2961		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
iostream	2955		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
`( * stdioflush ) ( )`	2950	`assign_private`	2941
`bp`	2941	`delbuf`	2941
`isfx_special`	2942	`ispecial`	2942
`osfx_special`	2942	`ospecial`	2942
`state`	2942	`x_fill`	2947
`x_flags`	2942	`x_precision`	2947
`x_tie`	2942	`x_width`	2947

IFile

This class provides a surrogate object for a file.

Do not derive your own classes from this class.

Class header file: ifilesys.hpp

IFile - Hierarchy List

IMStreamable

IFileSystemEntity

IFile

IFile - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct objects of this class.

~IFile

```
public:virtual ~IFile()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IFile

Constructs an IFile object.

Overload 1

```
public:IFile(const IFile& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the copy constructor to create an IFile object copied from the given one.

Overload 2

```
public:IFile()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Assignment Operator

Use this operator to replace the existing object with the one specified.

operator =

```
public:IFile& operator =(const IFileSystemEntity& right)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The assignment operator.

copy If this entity refers to a physical file, the IFile now refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a file.

Class Constants

The kind of the file system entity. A form of homegrown RTTI.

kKind

```
public:static const IFileKind kKind
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

An IFileKind constant that identifies the IFile class. This constant is used for dynamic type checking.

Setting and Getting File Values

Use these member functions to set and get a file's size, get a file's modification time, or to create a temporary file.

createTemporary

Creates a temporary file.

Overload 1

```
public:static IFile
  createTemporary( const ICCSId& id,
                   const IFileName& name = uniqueName ( ),
                   const IFileSize initialSize = 0u )
```

This is supported on **AIX OS/400 z/OS**

Creates a temporary file in the appropriate place for temporary storage on the current system. The file is only ″temporary″ in that sense; it is not automatically deleted when the IFile object is destroyed.

id The code page, or character set id, with which to open the file. If the value is 0, the code page or coded character set id of the application will be used.

name The name for the temporary file. If none is provided, a unique name is generated.

initialSize
The initial size of the created file, in bytes. If none is specified, the file is empty.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:static IFile
  createTemporary( const IFileName& name = uniqueName ( ),
                   const IFileSize initialSize = 0u )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a temporary file in the appropriate place for temporary storage on the current system. The file is only "temporary" in that sense; it is not automatically deleted when the IFile object is destroyed.

name The name for the temporary file. If none is provided, a unique name is generated.

initialSize
The initial size of the created file, in bytes. If none is specified, the file is empty.

Exception

Exception	Description
IEntityExists	Thrown if a name is provided and a a temporary file with that name already exists.

getRecord

```
public:IFSFileRecord* getRecord() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file system record.

setModificationTime

```
public:void
  setModificationTime( const ITimeStamp& modificationTime ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the file's modification time. Subsequent calls to IFileSystemEntity::modificationTime return the new value.

setSize

```
public:void setSize(IFileSize newSize)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the file's size to the specified size. If the current size is larger, the file is truncated. If the current size is smaller, the file is padded with arbitrary data.

newSize
Desired size of the file, in bytes.

size

```
public:IFileSize size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the size of the file, in bytes.

Special Downcast Constructors

Use these constructors to create IFile objects based on the given information, such as a host path name, a partial path name, a file system entity, or a file system directory record.

IFile

Constructs an IFile object.

Overload 1

```
public:IFile( const IPathName& path,
       ECreateFlags flags = kMustExist )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IFile object from the given portable path name.

path The portable path name.

flags Specifies whether the constructor creates or overwrites a file on the disk.

Overload 2

```
public:IFile(const IFileSystemEntity& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IFile object that is constructed from a copy of the given file system entity, with type checking.

copy If this entity refers to a physical file, the construced IFile refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a file.

Overload 3

```
public:IFile( const IHostPathName& path,
       ECreateFlags flags = kMustExist,
       const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IFile object from a host-specific path name.

path Text string containing the host-specific path name.

flags Specifies whether the constructor creates or overwrites a file on the disk.

parser The path name parser which should be used to parse the provided path. If no parser is specified, the default for the current system is used.

Overload 4

```
public:IFile(IFSFileRecord* Record)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IFile object from the given file system record.

Streaming Support

Use this member function to create a stream.

createStream

Creates an IBM Open Class Library stream for I/O on this file.

Overload 1

```
public:IDataStream*
  createStream( const ICCSId& id,
                int modeForMe = in + out,
                int modeForOthers = 0,
                IDataStream::EStreamEncoding encoding = IDataStream::kInteroperabl
```

This is supported on **AIX OS/400 z/OS**

Creates an IBM Open Class Library stream for I/O on this file.

id
The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. See IFileIOStream::IFileIOStream for details.

modeForOthers
Specifies the modes in which you want to allow others to open the file concurrently. See IFileIOStream::IFileIOStream for details.

encoding
The format for data on the stream. See IDataStream::EStreamEncoding for details.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:IDataStream*
  createStream( int modeForMe = in + out,
                int modeForOthers = 0,
                IDataStream::EStreamEncoding encoding =
                IDataStream::kInteroperableBinary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an IBM Open Class Library stream for I/O on this file.

modeForMe
The mode in which the file should be opened. See IFileIOStream::IFileIOStream for details.

modeForOthers
Specifies the modes in which you want to allow others to open the file concurrently. See IFileIOStream::IFileIOStream for details.

encoding
The format for data on the stream. See IDataStream::EStreamEncoding for details.

IFile - Enumerations

EOpenMode

Enumerator that defines the mode with which a file is to be opened.

Variation 1

```
enum EOpenMode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 binary=0200 }
```

This is supported on **WIN OS/2 AIX z/OS**

Variation 2

```
enum EOpenMode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 binary=0200,
                 text=0400 }
```

This is supported on **OS/400**

IFile - Inherited Member Functions and Data

Inherited Public Functions

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileSystemEntity	281	connected	286
copyTo	284	defaultParser	283
deleteSelf	284	getRecord	282
hostPath	282	IFileSystemEntity	281
isA	283	modificationTime	282
moveTo	284	name	282
operator !=	286	operator =	282
operator ==	286	parent	285
path	283	readFromStream	285
setName	283	temporary	283
topmostParent	285	uniqueName	284
valid	286	volume	285
writeToStream	285		

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable writeToStream	200 202	readFromStream	201

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
setInvalid	286	typeSafeAssign	283

Inherited Protected Data

None

IFileOperation

This is an abstract base class for recursive file system operations.

IFileOperation provides virtual member functions which may be overridden to provide customized error handling and progress indicators

Do not derive your own classes from this class.

Class header file: ifilesys.hpp

IFileOperation - Hierarchy List

IFileOperation

IFileSystemCopier

IFileSystemMover

IFileOperation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct file operation objects.

~IFileOperation

```
public:virtual ~IFileOperation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IFileOperation

Constructor for use by subclasses.

Overload 1

```
protected:IFileOperation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IFileOperation( EFailureAction action,
                          unsigned int tellMeTimes )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

action What to do if the operation encounters an error.

tellMeTimes

The number of times that reportProgress should be called during the operation.

Accessing Member Functions

The member functions in this group provide access to member functions.

failureAction

```
public:EFailureAction failureAction() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the behavior of an operation when an unhandled error occurs.

reportFrequency

```
public:unsigned int reportFrequency() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the progress report frequency for this operation. The frequency is the number of times that reportProgress is called during the operation.

setFailureAction

```
public:void setFailureAction(EFailureAction actionOnFailure)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the behavior of an operation when an unhandled error occurs.

Callback Functions

Use these callback member functions to handle periodic or special operations.

handleFailure

```
public:virtual bool
  handleFailure( const IException& reason,
                 const IFileSystemEntity& current )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called when a failure other than a name conflict occurs.

This member function is called if the operation fails for some reason other than a name conflict. The exception is passed in, along with the entity that caused the failure.

reason The exception that caused the failure.

current

The entity being operated upon when the failure occurred.

Return

true to retry the operation on the current entity or false to simply throw the exception. This implementation returns false. Even if an exception is thrown, the operation first attempts to process all remaining entities if failureAction is set to kContinue.

renameNeeded

```
public:virtual bool
  renameNeeded( IFileName& modifyThisName,
                const IFileSystemEntity& currentEntity )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called whenever a name conflict arises during the operation. Developers may override this and provide a mechanism for resolving name conflicts, e.g. by prompting the user for a new name.

modifyThisName
The name of the entity that resulted in a conflict. To continue the operation, set this parameter to a new name and return true.

currentEntity
The entity currently being operated upon.

Return
true to continue the operation or false to terminate it. This implementation returns false. If true is returned, the modifyThisName parameter must be set to a new name.

reportProgress

```
public:virtual bool
  reportProgress( double fractionDone,
                  const IFileSystemEntity& currentEntity )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called to report the progress of this operation. As the operation progresses, reportProgress is called as many times as specified in the tellMeTimes argument to the constructor.

fractionDone
The portion of the operation that has been completed, expressed as a floating point number between 0 and 1.

currentEntity
The entity currently being operated upon. You may want to show its name in your progress indicator.

Return
true to continue the operation or false to terminate it. This implementation returns true.

Subclass Interface

These member functions are provided to be overridden by subclasses.

doOneDirectory

```
protected:virtual IDirectory
  doOneDirectory( const IDirectory& dir,
                  IDirectory& dest,
                  const IFileName& newName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Callback to perform the operation on a single directory.

dir The directory on which to operate.

dest The target directory for this entity. In an operation like move or copy, this is the destination to which the directory should be moved or copied. Other operations may choose to ignore this parameter.

newName
If non-empty, the new name desired for the directory in its target location.

Return
An IDirectory pointing to the new location of the original directory

(e.g. the moved or copied directory) If this operation does not affect the directory's location, the same IDirectory should be returned.

Subclassing

This implementation of doOneDirectory simply creates a new directory in the destination location that has the same attributes as the original directory. If your subclass performs an operation for which this is not appropriate, you may override this to do something else. If you need to create a new directory and also perform some other action, write an override which calls up to this member function.

doOneEntity

```
protected:virtual IFileSystemEntity
  doOneEntity( const IFileSystemEntity& entity,
               IDirectory& target,
               const IFileName& newName ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Callback to perform the operation on a single file.

Subclasses must override this function to perform whatever action is desired for each entity affected by the operation. For example, a move operation uses this function to move the file to the destination directory.

entity The entity on which to operate.

target The target directory for this entity. In an operation like move or copy, this is the destination to which the file should be moved or copied. Other operations may choose to ignore this parameter.

newName

If non-empty, the new name desired for the entity in its target location.

Return

An entity pointing to the new location of the original entity (e.g. the moved or copied object). If this operation does not affect the entity's location, the same entity should be returned.

execute

```
protected:IFileSystemEntity
  execute( const IFileSystemEntity& entity,
           IDirectory& target,
           const IFileName& newName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Member function called by subclasses to start the operation.

This member function calls the doOneDirectory function for every directory at or under the specified entity. It calls the doOneFile function for every file at or under entity. Depending on the arguments passed to the constructor, it also does error handling and progress reporting.

entity The entity on which to operate.

target The target directory, e.g. the destination directory for a move or copy operation.

newName

If non-empty, the new name desired for the entity in its target location.

Return

An entity pointing to the new location of the original entity (e.g.

the moved or copied object). If this operation does not affect the entity's location, the same entity is returned.

postProcess

```
protected:virtual void
  postProcess( IFileSystemEntity& source,
               IFileSystemEntity& result )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Callback to clean up after the operation.

This is called after doOneEntity has been called for an entity and all of its children. It can be used to perform any necessary cleanup actions, such as deleting the original entity after a move operation.

source The original entity that was operated upon.

result The entity that was a result of the operation, i.e. that was returned from execute.

IFileOperation - Enumerations

EFailureAction

Specifies the behavior of an operation when an unhandled error occurs.

kStop Throws the exception immediately.

kContinue

Attempts to complete the operation on all remaining entities, and only then throws the exception.

IFileOperation - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IFileOperationCancelled

IFileOperationCancelled is an exception that indicates that a user cancelled a file system operation.

This exception is thrown if the reportProgress member function in an IFileOperation subclass returns false.

Class header file: iexcfile.hpp

IFileOperationCancelled - Hierarchy List

IException

IFileOperationCancelled

IFileOperationCancelled - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IFileOperationCancelled object from another.

~IFileOperationCancelled

```
public:virtual ~IFileOperationCancelled()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IFileOperationCancelled

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IFileOperationCancelled(const IFileOperationCancelled&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IFileOperationCancelled(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function constructs an exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IFileOperationCancelled - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IFileStdIOStream

Standard-library-style stream class which can be construced from an IFile.

This class allows ANSI iostream-based code to interoperate with the IBM Open Class file system API. IFileStdIOStream class consists mostly of constructors and open/close member functions. To perform I/O operations, use the member functions inherited from iostream.

NOTE: IFileIOStream is a typedef of either IFileUslIOStream or IFileStdIOStream. When __IOC_ANSI_STREAM is defined, IFileIOStream is a typedef of IFileStdIOStream.

Class header file: ifilestr.hpp

IFileStdIOStream - Hierarchy List

IFileStdIOStream

IFileStdIOStream - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IFileStdIOStream

```
public:virtual ~IFileStdIOStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class. If the stream is still open, the destructor attempts to close it. However, it does not throw an exception if the close fails.

IFileStdIOStream

Constructs an IFileStdIOStream object.

Overload 1

```
public:IFileStdIOStream( const IFile& fileToOpen,
                         const ICCSId& id,
                         int modeForMe = std::ios::in + std::ios::out,
                         int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:IFileStdIOStream( const IHostPathName& path,
                         const ICCSId& id,
                         int modeForMe = std::ios::in + std::ios::out,
                         int modeForOthers = 0,
                         const ICompoundNameParser& = IFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the file which should be opened.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 3

```
public:IFileStdIOStream( const IFile& fileToOpen,
                         int modeForMe = std::ios::in + std::ios::out,
                         int modeForOthers = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

fileToOpen
The file to be opened for streaming.

modeForMe

The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers

Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Overload 4

```
public:IFileStdIOStream( const IHostPathName& path,
                         int modeForMe = std::ios::in + std::ios::out,
                         int modeForOthers = 0,
                         const ICompoundNameParser& = IFile::defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

path A host-specific path name to the file which should be opened.

modeForMe

The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers

Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

parser

The path name parser which should be used to parse the

host-specific path. If none is specified, the default parser for the current system is used.

Member Access

Use these member functions to access the data of this class.

file

```
public:IFile file() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file to which this stream is attached.

Opening and Closing Streams

Use these member functions to open and close the stream.

close

```
public:void close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Closes the stream if it is open. Unlike the destructor, this member function does throw an exception if it fails.

open

Opens the stream.

Overload 1

```
public:void
  open( const IFile& fileToOpen,
        const ICCSId& id,
        int modeForMe = std::ios::in + std::ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another file. If the stream is already open, it closes current file first.

fileToOpen
The file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:void
  open( const IFile& fileToOpen,
        int modeForMe = std::ios::in + std::ios::out,
        int modeForOthers = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attaches the stream to another file. If the stream is already open, it closes current file first.

fileToOpen
The file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- std::ios::in - Opens the file for reading.
- std::ios::out - Opens the file for writing.
- std::ios::app - Append mode - seeks to end before all writes.
- std::ios::ate - At end - seeks to the end just after opening.
- std::ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- std::ios::in - Others may read the file.
- std::ios::out - Others may write to the file.

IFileStdIOStream - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IFileSystemCopier

This class copies a file system entity. If the file system entity is a directory, it copies all of the entity's children.

Subclassing

Developers who want to perform copies with progress reports, name conflict resolution, or customized error handling must create a subclass and override one or more of the callback member functions declared in IFileOperation.

Class header file: ifilesys.hpp

IFileSystemCopier - Hierarchy List

IFileOperation

IFileSystemCopier

IFileSystemCopier - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct file system copier objects.

~IFileSystemCopier

```
public:virtual ~IFileSystemCopier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys a file system copier object.

IFileSystemCopier

```
public:IFileSystemCopier( EFailureAction action = kStop,
                  unsigned int tellMeTimes = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a file system copier object.

action What to do if the operation encounters an error. See IFileSystemCopier::EFailureAction for details.

tellMeTimes
The number of times that reportProgress should be called during the copy operation.

Callback Functions

These callback functions are overridden by derived classes to handle notifications from the base class.

doOneEntity

```
protected:virtual IFileSystemEntity
  doOneEntity( const IFileSystemEntity& entity,
               IDirectory& destination,
               const IFileName& newName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called for each entity to be copied.

Copy Operations

This function implements the copier operation.

copy

```
public:IFileSystemEntity
  copy( const IFileSystemEntity& source,
        IDirectory& destination,
        const IFileName& newName = "" )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies an entity to a new parent directory. If the entity is a directory, all of its children are copied as well. The error handling and progress reporting behavior of the copy depends on the arguments that were passed to this object's constructor.

source The entity to be copied.

destination
The directory to which this entity should be copied.

newName
An optional new name for the resulting entity. If no new name is provided, the current name is used.

IFileSystemCopier - Inherited Member Functions and Data

Inherited Public Functions

IFileOperation			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileOperation	267	failureAction	268
handleFailure	268	renameNeeded	268
reportFrequency	268	reportProgress	269
setFailureAction	268		

Inherited Public Data

None

Inherited Protected Functions

IFileOperation			
Definition	**Page Number**	**Definition**	**Page Number**
doOneDirectory	269	doOneEntity	270
execute	270	IFileOperation	267
postProcess	271		

Inherited Protected Data

None

IFileSystemEntity

This is the base class for all physical file system entities and objects.

Do not derive your own classes from this class.

Class header file: ifilesys.hpp

IFileSystemEntity - Hierarchy List

IMStreamable

IFileSystemEntity

IFifoFile

INativeObject

IDirectory

IFile

IFileSystemEntity - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct objects of this class.

~IFileSystemEntity

```
public:virtual ~IFileSystemEntity()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class.

IFileSystemEntity

Constucts a file system entity.

Overload 1

```
public:IFileSystemEntity( const IHostPathName& hostPath,
                          const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a file system entity from a host-specific path name to a file system object.

parser The path name parser which should be used to format the returned host-specific path. If no parser is specified, the default for the current system is used.

Overload 2

```
public:IFileSystemEntity(const IPathName& path)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an entity from a portable path name to a file system object.

Overload 3

```
public:IFileSystemEntity(const IFileSystemEntity& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a file system entity that is copied from the given one.

Overload 4

```
public:IFileSystemEntity(IFSRecord* Record)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a file system entity from a file system record.

Overload 5

```
public:IFileSystemEntity()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an empty file system entity object.

operator =

```
public:virtual IFileSystemEntity&
  operator =( const IFileSystemEntity& right )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign the given file system entity.

Accessing File System Properties and Entities

Use these member functions to access file system entities and obtain information about them. For example, you can obtain this information: the file system entity name, a portable path name for the entity, the time when the entity was last modified, the host path name, whether the entity is temporary, and the file system record for the entity. You can also set the current file system entity to the given one.

getRecord

```
public:IFSRecord* getRecord() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file system record.

hostPath

```
public:IHostPathName
  hostPath( const ICompoundNameParser& parser = defaultParser ( ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a host path name, using the specified parser.

modificationTime

```
public:ITimeStamp modificationTime() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the time of the most recent modification to the on-disk entity.

Windows Considerations

On Windows 95, this method for IVolume is not supported and it throws an IAccessError exception.

name

```
public:IFileName name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of this file system entity.

path

```
public:IPathName path() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a portable path name for this entity.

setName

Changes the name of this file system entity.

```
public:void setName(const IFileName& newName) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the name of the current file system entity to the given one.

temporary

```
public:bool temporary() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks whether this entity is temporary.

This does not check to see whether the entity resides in any particular system temporary directory; it simply returns true if the entity was created with IFile::createTemporary.

Checking the Type of a File System Entity

Use these member functions to verify the file system entity type.

isA

```
public:bool isA(const IFileKind& kind) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks whether this entity is of the specified type.

kind The type of entity to check for.

typeSafeAssign

```
protected:void
  typeSafeAssign( const IFileSystemEntity& right,
                  const IFileKind& kind )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Class Constants

Constants that are nested within the IFileSystemEntity class.

defaultParser

Returns the default path name parser for this operating system.

```
public:static const ICompoundNameParser& defaultParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the default path name parser for this operating system. This member function is a syntactic convenience; it always returns the same parser as IPathName::defaultParser.

uniqueName

```
public:static const IFileName& uniqueName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a unique name to be used for a file system entity.

Manipulating File System Entities

Use these member functions to move, copy, and delete file system entities.

copyTo

```
public:IFileSystemEntity
  copyTo( IDirectory& destination,
          const IFileName& newName = "",
          EFailureAction action = kStop ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the entity to a different parent directory. If the entity is a directory or volume, all of its children are copied too.

destination
The directory to which this entity should be copied.

newName
An optional new name for the new copy of the entity. If no new name is provided, the current name is used.

action What to do if an error occurs during the copy.

deleteSelf

```
public:void deleteSelf(bool deleteChildren = false)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes this entity and, optionally, its children.

deleteChildren
If this parameter is true and the entity is a directory, all entities within the directory are deleted too. If the entity is a non-empty directory and this parameter is false, IMustBeEmpty is thrown.

Exception

Exception	Description
IMustBeEmpty	Attempted to delete a non-empty directory without specifying deleteChildren = true.
IEntityInUse	Attempted to delete an entity that is open or otherwise in use.

getRecord

```
public:IFSRecord* getRecord() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file system record.

moveTo

```
public:void
  moveTo( IDirectory& destination,
          const IFileName& newName = "",
          EFailureAction action = kStop )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the entity to a different parent directory.

destination
The directory to which this entity should be moved.

newName
An optional new name for this entity after it is moved. If no new name is provided, the current name is used.

action What to do if an error occurs during the move.

Navigating the File System Directory

Use these member functions to navigate the directory hierarchy.

parent

```
public:IDirectory parent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the directory in which this entity resides. If the entity has no parent, that is, it is a root directory, an invalid entity is returned.

topmostParent

```
public:IDirectory topmostParent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the root of the directory tree in which this entity resides. If the entity has no parent, e.g. it is a root directory, an invalid entity is returned.

volume

```
public:IVolume volume() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the volume on which this entity resides.

Streaming Objects In and Out

Use these member functions to stream file system entities in or out using the given stream.

readFromStream

```
public:virtual void readFromStream(IDataStream& fromwhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads this entity object from an IBM Open Class stream.

Generally you should use the global operator<<=, which will in turn call this method, rather than using this method directly.

fromWhere
The stream to use to stream in the object.

writeToStream

```
public:virtual void writeToStream(IDataStream& towhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes this entity object to an IBM Open Class stream.

Generally you should use the global operator>>=, which will in turn call this method, rather than using this method directly.

toWhere

The stream to use to stream out the object.

Testing and Comparing File System Entity Objects

Use these member functions and operators to test and compare file system entities. For example, you can test two objects for equality, whether two entities point to the same physical object, and whether the current file system entity represents a physical, on-disk entity.

connected

```
public:bool connected() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Test whether this object represents a physical, on-disk entity.

This member function goes to the disk to verify that the object points to a valid physical object on the file system.

operator !=

```
public:bool operator !=(const IFileSystemEntity& object) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests the current file system entity and the given one for equality using the inequality operator.

operator ==

```
public:bool operator ==(const IFileSystemEntity& object) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether two entities point to the same physical object on the file system, even if they were constructed from different path names.

valid

```
public:bool valid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether this object represents an entity.

Returns false for entity objects that were created with the default constructor and for objects that have been deleted with deleteSelf.

This member function does NOT check to see whether the entity points to a valid physical object on the disk. To do so, use connected instead.

setInvalid

```
protected:void setInvalid()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Renders this file system entity invalid.

IFileSystemEntity - Enumerations

ECreateFlags

Specifies whether constructors create entities when needed.

kMustExist
Throw IObjectNotFound if the entity does not already exist.

kCreateIfNeeded
Create the disk entity if it does not already exist.

kCreateOnly
Always create; throw IAlreadyExists if it already exists.

kReplaceExisting
If the entity already exists, delete it and create a new one.

EFailureAction

Specifies the behavior of a move or copy operation when an error occurs.

IFileSystemEntity - Inherited Member Functions and Data

Inherited Public Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable	200	readFromStream	201
writeToStream	202		

Inherited Protected Data

None

IFileSystemMover

Moves a file system entity. Recursively moves all of the entity's children if it is a directory.

Subclassing

Developers who want to perform moves with progress reports, name conflict resolution, or customized error handling must create a subclass and override one or more of the callback member functions declared in IFileOperation.

Class header file: ifilesys.hpp

IFileSystemMover - Hierarchy List

IFileOperation

IFileSystemMover

IFileSystemMover - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct file system mover objects.

~IFileSystemMover

```
public:virtual ~IFileSystemMover()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys a file system mover object.

IFileSystemMover

```
public:IFileSystemMover( EFailureAction action = kStop,
                         unsigned int tellMeTimes = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a file system mover object.

Use this constructor to create a file system mover object that specifies error handling and the number of times that reportProgress should be called during the operation.

action What to do if the operation encounters an error. See IFileSystemCopier::EFailureAction for details.

tellMeTimes
The number of times that reportProgress should be called during the copy operation.

Callback Functions

These callback functions are overridden by derived classes to handle notifications from the base class.

doOneEntity

```
protected:virtual IFileSystemEntity
  doOneEntity( const IFileSystemEntity& entity,
               IDirectory& destination,
               const IFileName& newName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called for each entity to be moved.

postProcess

```
protected:virtual void
  postProcess( IFileSystemEntity& source,
               IFileSystemEntity& result )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called after an entity has been moved, to provide post processing.

Moving a File System Entity

Use this member function to implement the mover operations.

move

```
public:IFileSystemEntity
  move( IFileSystemEntity& source,
        IDirectory& destination,
        const IFileName& newName = "" )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves an entity to a new parent directory. If the entity is a directory, all of its children are moved as well. The error handling and progress reporting behavior of the move depends on the arguments that were passed to this object's constructor.

source The entity to be moved.

destination
The directory to which this entity should be moved.

newName
An optional new name for the entity after it is moved. If no new name is provided, the current name is used.

IFileSystemMover - Inherited Member Functions and Data

Inherited Public Functions

IFileOperation			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileOperation	267	failureAction	268
handleFailure	268	renameNeeded	268
reportFrequency	268	reportProgress	269
setFailureAction	268		

Inherited Public Data

None

Inherited Protected Functions

IFileOperation			
Definition	**Page Number**	**Definition**	**Page Number**
doOneDirectory	269	doOneEntity	270
execute	270	IFileOperation	267

Inherited Protected Data

None

IFileUslIOStream

Standard-library-style stream class which can be construced from an IFile.

This class allows iostream-based code to interoperate with the IBM Open Class file system API. IFileUslIOStream class consists mostly of constructors and open/close member functions. To perform I/O operations, use the member functions inherited from iostream.

NOTE: IFileIOStream is a typedef of either IFileUslIOStream or IFileStdIOStream.

When the __IOC_ANSI_STREAM is not defined, IFileIOStream is a typedef of IFileUslIOStream.

Class header file: ifilestr.hpp

IFileUslIOStream - Hierarchy List

ios

istream

iostream

IFileUslIOStream

IFileUslIOStream - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IFileUslIOStream

```
public:virtual ~IFileUslIOStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys and object of this class. If the stream is still open, the destructor attempts to close it. However, it does not throw an exception if the close fails.

IFileUslIOStream

Constructs an IFileStdIOStream object.

Overload 1

```
public:IFileUslIOStream( const IHostPathName& path,
                         int modeForMe = ios::in + ios::out,
                         int modeForOthers = 0,
                         const ICompoundNameParser& = IFile::defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

path A host-specific path name to the file which should be opened.

modeForMe

The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers

Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.

- ios::in - Others may read the file.
- ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Overload 2

```
public:IFileUslIOStream( const IFile& fileToOpen,
                int modeForMe = ios::in + ios::out,
                int modeForOthers = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

fileToOpen
The file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.
- ios::out - Others may write to the file.

Overload 3

```
public:IFileUslIOStream( const IHostPathName& path,
                const ICCSId& id,
                int modeForMe = ios::in + ios::out,
                int modeForOthers = 0,
                const ICompoundNameParser& = IFile::defaultParser ( ) )
```

This is supported on **AIX OS/400 z/OS**

path A host-specific path name to the file which should be opened.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.
- ios::out - Others may write to the file.

parser
The path name parser which should be used to parse the host-specific path. If none is specified, the default parser for the current system is used.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 4

```
public:IFileUslIOStream( const IFile& fileToOpen,
                  const ICCSId& id,
                  int modeForMe = ios::in + ios::out,
                  int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

fileToOpen
The file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the mode you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.

- ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Member Access

Use these member functions to access the data of this class.

file

```
public:IFile file() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file to which this stream is attached.

Open and Close Functions

Use these member functions to open and close the stream.

close

```
public:void close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

open

Opens the stream.

Overload 1

```
public:void
  open( const IFile& fileToOpen,
        int modeForMe = ios::in + ios::out,
        int modeForOthers = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attaches the stream to another file. If the stream is already open, it closes current file first.

fileToOpen
The file to be opened for streaming.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful:

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.
- ios::out - Others may write to the file.

Overload 2

```
public:void
  open( const IFile& fileToOpen,
        const ICCSId& id,
        int modeForMe = ios::in + ios::out,
        int modeForOthers = 0 )
```

This is supported on **AIX OS/400 z/OS**

Attaches the stream to another file. If the stream is already open, it closes current file first.

fileToOpen
The file to be opened for streaming.

id The code page, or character set id, with which to open the stream. If the value is 0, the code page or coded character set id of the application will be used.

modeForMe
The mode in which the file should be opened. Values for this parameter are formed by OR-ing together constants from the ios::open_mode enum. The following values are useful):

- ios::in - Opens the file for reading.
- ios::out - Opens the file for writing.
- ios::app - Append mode - seeks to end before all writes.
- ios::ate - At end - seeks to the end just after opening.
- ios::trunc - Truncates an existing file to 0 length when opening.

modeForOthers
Specifies the modes you want to allow others to open the file concurrently with you. This is formed by OR-ing values from ios::open_mode. Only the following are useful:

- 0 - Others are not allowed to read or write.
- ios::in - Others may read the file.
- ios::out - Others may write to the file.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

IFileUsIIOStream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941

ios			
Definition	**Page Number**	**Definition**	**Page Number**
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
~iostream	2955	iostream	2955

istream			
Definition	**Page Number**	**Definition**	**Page Number**
~istream	2961	gcount	2962
get	2962	get_complicated	2966
getline	2966	ignore	2969
ipfx	2984	isfx	2985
istream	2961	operator >>	2972
peek	2970	putback	2983
read	2970	rs_complicated	2971
seekg	2983	sync	2984
tellg	2984		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

istream			
Definition	**Page Number**	**Definition**	**Page Number**
do_ipfx	2985	eatwhite	2972

istream			
Definition	**Page Number**	**Definition**	**Page Number**
istream	2961		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
iostream	2955		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

INativeObject

This class provides a surrogate object for a file system entity which is not a standard UNIX file system entity.

OS/400 Considerations

On OS/400 the file system classes interact with the Integrated File System (IFS). The root directory of the IFS is '/'. A sub-directory of the root directory is the data management file system (QSYS.LIB). QSYS.LIB is the original or native file system of OS/400. The INativeObject class represents objects from the QSYS.LIB file system. On OS/400 objects have a native type, for instance an executable is typically called a **program object** which has a native type of ***PGM** .

Class header file: ifilesys.hpp

INativeObject - Hierarchy List

IMStreamable

IFileSystemEntity

INativeObject

INativeObject - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct objects of this class.

~INativeObject

```
public:virtual ~INativeObject()
```

This is supported on **OS/400**

Destroys and object of this class.

INativeObject

Constructs an INativeObject.

Overload 1

```
public:INativeObject(const IFileSystemEntity& copy)
```

This is supported on **OS/400**

Use this copy constructor to create an INativeObject that is constructed from a copy of the given file system entity, with type checking.

copy If this entity refers to a native object, the constructed INativeObject refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a native object.

Overload 2

```
public:INativeObject( const IPathName& path,
                      ECreateFlags flags = kMustExist )
```

This is supported on **OS/400**

Use this constructor to create an INativeObject from the given portable path name.

path The portable path name.

flags Specifies whether the constructor creates or overwrites a native object on the disk.

Overload 3

```
public:INativeObject( const IHostPathName& path,
                      ECreateFlags flags = kMustExist,
                      const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **OS/400**

Use this constructor to create an INativeObject object from a host-specific path name.

path Text string containing the host-specific path name.

flags Specifies whether the constructor creates or overwrites a native object on the disk.

parser The path name parser which should be used to parse the provided path. If no parser is specified, the default for the current system is used.

Overload 4

```
public:INativeObject(IFSNativeRecord* Record)
```

This is supported on **OS/400**

Use this constructor to create an INativeObject object from a file system record.

Overload 5

```
public:INativeObject(const INativeObject& copy)
```

This is supported on **OS/400**

Use this copy constructor to create an INativeObject copied from the given one.

Overload 6

```
public:INativeObject()
```

This is supported on **OS/400**

The default constructor.

Assignment Operator

Use this operator to replace the existing object with the one specified.

operator =

```
public:INativeObject& operator =(const IFileSystemEntity& right)
```

This is supported on **OS/400**

Assigns an INativeObject to an entity object, with type checking.

right If this entity refers to a physical native object, the INativeObject now refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a native object.

Class constants

The kind of the file system entity. A form of homegrown RTTI.

kKind

```
public:static const IFileKind kKind
```

This is supported on **OS/400**

An IFileKind constant that identifies the INativeObject class. This constant is used for dynamic type checking.

Setting and Getting File Values

Use these member functions to set and get a native object's attributes.

getRecord

```
public:IFSNativeRecord* getRecord() const
```

This is supported on **OS/400**

Returns the file system record of this object.

nativeType

```
public:IText nativeType() const
```

This is supported on **OS/400**

Returns the type of the native object.

OS/400 Considerations

On OS/400 the native object type is returned. For example a program object would return a type of ***PGM**

setModificationTime

```
public:void
  setModificationTime( const ITimeStamp& modificationTime ) const
```

This is supported on **OS/400**

Sets the native object's modification time. Subsequent calls to IFileSystemEntity::modificationTime return the new value.

size

```
public:IFileSize size() const
```

This is supported on **OS/400**

Returns the size of the native object, in bytes.

INativeObject - Inherited Member Functions and Data

Inherited Public Functions

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileSystemEntity	281	connected	286
copyTo	284	defaultParser	283
deleteSelf	284	getRecord	282
hostPath	282	IFileSystemEntity	281
isA	283	modificationTime	282
moveTo	284	name	282
operator !=	286	operator =	282
operator ==	286	parent	285
path	283	readFromStream	285
setName	283	temporary	283

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
topmostParent valid writeToStream	285 286 285	uniqueName volume	284 285

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable internalizeFromStream	200 201	externalizeToStream typeRepresentation	201 201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable writeToStream	200 202	readFromStream	201

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
setInvalid	286	typeSafeAssign	283

Inherited Protected Data

None

IOS400PathNameParser

IOS400PathNameParser parses and formats OS/400 style pathnames.

Class header file: ipathnam.hpp

IOS400PathNameParser - Hierarchy List

ICompoundNameParser

IOS400PathNameParser

IOS400PathNameParser - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct objects of this class.

~IOS400PathNameParser

```
public:virtual ~IOS400PathNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IOS400PathNameParser

Constructs an IOS400PathNameParser object.

Overload 1

```
public:IOS400PathNameParser(const IText& type = kPgm)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default object type is kPgm, which sets the object type to *PGM.

type Object type to be set.

Overload 2

```
public:IOS400PathNameParser(const IOS400PathNameParser& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

copyMe

The object to copy.

operator =

```
public:IOS400PathNameParser&
  operator =( const IOS400PathNameParser& copyMe )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IOS400PathNameParser object to another.

copyMe

The object to copy.

Accessing Object Types

Use these member functions to access the object type set for IOS400PathNameParser.

objectType

```
public:IText objectType() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the object type set for IOS400PathNameParser.

setObjectType

```
public:void setObjectType(const IText& type)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes the object type for IOS400PathNameParser. This member function is called when parsing another pathname with a different object type.

Assessing the Separator

Use this member function to obtain the separator character used to separate path name components.

separator

```
public:virtual ioc::unichar_t separator() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the UniCode character that is used to separate path name components.

Parsing and Formatting

Use these member functions to parse and format path names.

formatName

```
public:virtual IHostPathName
  formatName( const ICompoundName& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts an ICompoundName into a platform-specific text string.

parseName

```
public:virtual ICompoundName parseName(const IHostPathName&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts a platform-specific text string into an ICompoundName object.

You can specify the library name with the following values:
- *CURLIB - current library
- *LIBL - library list
- *USRLIBL - user library list
- *ALLUSR - all user libraries
- *ALL - all libraries

When any of these values is used, it will be replaced by the actual library name if the object exists. Otherwise, an exception will be thrown.

Typedefs and Constant Definitions

Type definitions and constants defined by this class.

kAll

```
public:static const IText kAll
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kAll sets the object type to *ALL.

When the object type is set to *ALL, All objects with the same given name will be filtered out. The object type of the first occurrence in the list will then be used to construct the IHostPathName object.

kBndDir

```
public:static const IText kBndDir
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kBndDir sets the object type to *BNDDIR.

kCmd

```
public:static const IText kCmd
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kCmd sets the object type to *CMD.

kDtaQ

```
public:static const IText kDtaQ
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kDtaQ sets the object type to *DTAQ.

kFile

```
public:static const IText kFile
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kFile sets the object type to *FILE.

kLib

```
public:static const IText kLib
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kLib sets the object type to *LIB.

kLocale

```
public:static const IText kLocale
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kLocale sets the object type to *LOCALE.

kModule

```
public:static const IText kModule
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kModule sets the object type to *MODULE.

kMsgF

```
public:static const IText kMsgF
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kMsgF sets the object type to *MSGF.

kMsgQ

```
public:static const IText kMsgQ
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kMsgQ sets the object type to *MSGQ.

kPgm

```
public:static const IText kPgm
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kPgm sets the object type to *PGM.

kSrvPgm

```
public:static const IText kSrvPgm
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This static constant is used by IOS400PathNameParser::IOS400PathNameParser and IOS400PathNameParser::setObjectType member functions to define the object type for the instance. kSrvPgm sets the object type to *SRVPGM.

IOS400PathNameParser - Inherited Member Functions and Data

Inherited Public Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
~ICompoundNameParser	226	formatName	227
operator !=	226	operator ==	227
parseName	227	separator	226

Inherited Public Data

None

Inherited Protected Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
ICompoundNameParser typeOf	226 227	operator =	226

Inherited Protected Data

None

IPathName

IPathName represents full or partial paths to file system entities.

This class allows clients to specify file system entities via path names. For example, constructing an IPathName as follows:

```
IPathName aPathName("/home/widget/work/Foo.C", IUnixPathNameParser());
```

allows the client to obtain an IFileSystemEntity representing the file Foo.C as follows:

```
IFile anEntity(aPathName);
```

You may provide a ″parser″ for the host-specific paths that you pass in, or you may use the default parser for the system on which your program is running. A parser that works for the system on which your program is running is available from IPathName::defaultParser().

Class header file: ipathnam.hpp

IPathName - Hierarchy List

IMStreamable

ICompoundName

IPathName

IPathName - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct objects of this class.

~IPathName

```
public:virtual ~IPathName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IPathName

Constructs an IPathName object.

Overload 1

```
public:IPathName(const IPathName& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a copy of the given object.

copyMe
The object to copy.

Overload 2

```
public:IPathName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Overload 3

```
public:IPathName( const IHostPathName& path,
          const ICompoundNameParser& aParser = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the specified parser to construct a path name from the text. If no parser is specified, the system's default file name parser is used.

path The text representation of the path.

aParser
The parser used to convert the text into path name components. If none is specified, the default filename parser is used.

Overload 4

```
public:IPathName( const IDirectory& startHere,
          const IHostPathName& partialPath,
          const ICompoundNameParser& aParser = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IPathName by appending a relative path to a directory's path. It uses the specified parser to construct a path name from the text. If no parser is specified, this system's default file name parser is used.

startHere
The directory whose path is used as the start of the constructed path.

partialPath
The text representation of the partial path that is appended to the directory's path.

aParser
The parser used to convert the text into path name components. If none is specified, the default filename parser is used.

Overload 5

```
public:IPathName(const ICompoundName& downcast)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Downcasts the given object and then constructs a copy of it.

downcast
The object to copy.

Overload 6

```
public:IPathName( const IDirectory& startHere,
          const IPathName& partialPath )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IPathName by appending a relative path to a directory's path.

startHere
The directory whose path is used as the start of the constructed path.

partialPath
The text representation of the partial path that is appended to the directory's path.

operator =

```
public:IPathName& operator =(const IPathName& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IPathName object to another.

copyMe
The object to copy.

Member Access

Use these member functions to access data of this class.

defaultParser

```
public:static const ICompoundNameParser& defaultParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the default path name parser for the operating system.

If you have a user-supplied path, a command line argument, etc., this is the parser that you should use to parse it.

Streaming

Use these member functions to stream objects of this class in and out.

readFromStream

```
protected:virtual void readFromStream(IDataStream& fromwhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads the object from the given data stream.

fromwhere
From where to stream the data.

writeToStream

```
protected:virtual void writeToStream(IDataStream& towhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the object to the given data stream.

towhere
Where to stream the data.

IPathName - Inherited Member Functions and Data

Inherited Public Functions

ICompoundName			
Definition	**Page Number**	**Definition**	**Page Number**
~ICompoundName	221	append	223
appendComponent	223	componentAt	222
factorCommonRoot	224	hash	222
ICompoundName	221	isFullPath	222
lastComponent	222	numberOfComponents	222
operator !=	222	operator =	222
operator ==	223	setFullPath	223
truncate	224		

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable	200	readFromStream	201
writeToStream	202		

Inherited Protected Data

None

IRootDirectoryIterator

Iterates over all root directories on the system.

On a system such as OS/2 or Windows, each volume _is_ a root directory, and a root directory iterator returns the exact same entities as a volume iterator. However, on systems like Unix, there can be many volumes but only one root directory, so the root directory iterator only returns the single root directory "/".

This class is designed to mimic a C++ pointer, similar to the iterator classes in the ANSI/ISO C++ draft standard. When the C++ Standard Library is available, IDirectoryIterator will inherit from input_iterator<IDirectory>.

Typical usage is as follows:

```
for (IRootDirectoryIterator aDirectory; aDirectory; aDirectory++)
{
     DoSomething( *aDirectory );
}
```

Class header file: ifileitr.hpp

IRootDirectoryIterator - Hierarchy List

IRootDirectoryIterator

IRootDirectoryIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct root directory iterator objects.

~IRootDirectoryIterator

```
public:virtual ~IRootDirectoryIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IRootDirectoryIterator

Constructs an IRootDirectoryIterator object.

Overload 1

```
public:IRootDirectoryIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Overload 2

```
public:IRootDirectoryIterator(const IRootDirectoryIterator& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor.

Copying a root directory iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

operator =

```
public:IRootDirectoryIterator&
  operator =( const IRootDirectoryIterator& copy )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IRootDirectoryIterator object to another.

Assigning a root directory iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

Comparison and validation operators.

Use these member functions to compare or validate root directory iterators.

operator !=

```
public:bool operator !=(const IRootDirectoryIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Two root directory iterators compare as equal if they are both past the end of their iteration, or if one was copy-constructed from the other.

operator ==

```
public:bool operator ==(const IRootDirectoryIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Two root directory iterators compare as equal if they are both past the end of their iteration, or if one was copy-constructed from the other.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to validate the iterator.

This member function is often used in the termination condition of a for or while loop. Returns false when there are no more root directories over which to iterate.

Miscellaneous

Miscellaneous functions.

end

```
public:static const IRootDirectoryIterator& end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an iterator that represents the end of iteration.

Standard iteration functions

Use these operators to iterator through the directories.

operator *

```
public:IDirectory operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current entity in the iteration. This member function allows the iterator to be used as if it were a normal C++ pointer, in expressions like:

```
FunctionCall( *anIterator )
```

operator ++

Pre-increment operator used for expressions like ++iterator. Proceeds to the next entity in the iteration.

Overload 1

Pre-increment operator used for expressions like ++iterator.

```
public:IRootDirectoryIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRootDirectoryIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ->

Returns a pointer to the current root directory in the iteration.

```
public:IDirectory* operator ->()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function allows the iterator to be used as if it were a normal C++ pointer, in expressions like:

```
anIterator->name()
```

IRootDirectoryIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITimeStamp

Objects of ITimeStamp class represent a specific point in time. An ITimeStamp object can be created from an IDate object, an IDate and ITime object, or a value that represents the number of seconds from the reference date 01/01/2000 00:00:00. If the timestamp is referring to a point in time before the reference date, a negative value must be used.

Accuracy is to the second since the time stamp is stored in seconds from a base time. Since the seconds are stored in a double, the time span representable is grotesquely large. Even represented in millenia, the size of the number is so large as to not be worth reproducing here.

This class provides functions to obtain specific information about an ITimeStamp object, compare between ITimeStamp objects, and manipulate an ITimeStamp object.

Class header file: itmstamp.hpp

ITimeStamp - Hierarchy List

ITimeStamp

ITimeStamp - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to constuct, copy, and destruct timestamp objects.

ITimeStamp

You can construct objects of this class by:

- giving a number of seconds from the reference date 01/01/2000 00:00:00.
- giving an IDate object only.
- giving an IDate and an ITime objects.
- copying from another ITimeStamp object.

Overload 1

```
public:ITimeStamp(const ITimeStamp& aTimeStamp)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimeStamp object by copying another ITimeStamp object.

Overload 2

```
public:ITimeStamp(const IDate& aDate, const ITime& aTime)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITimeStamp object by supplying an IDate and an ITime obect.

Overload 3

```
public:ITimeStamp(const IDate& aDate)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an object of this class by giving an IDate object.

Overload 4

```
public:ITimeStamp(double seconds = 0.0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an object of this class by giving a number of seconds from the reference date 01/01/2000 00:00:00.

Comparisons

Use these member functions to compare one timestamp to another for equality or relative magnitude.

operator !=

```
public:bool operator !=(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the ITimeStamp objects represent different timestamp.

operator <

```
public:bool operator <(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the left-hand operand represents a timestamp prior to the timestamp represented by the right-hand operand.

operator <=

```
public:bool operator <=(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the left-hand operand represents a timestamp prior to or identical to the timestamp represented by the right-hand operand.

operator ==

```
public:bool operator ==(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the ITimeStamp objects represent the same timestamp.

operator >

```
public:bool operator >(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the left-hand operand represents a timestamp subsequent to the timestamp represented by the right-hand operand.

operator >=

```
public:bool operator >=(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the left-hand operand represents a timestamp subsequent to or identical to the timestamp represented by the right-hand operand.

Constants

Class constants.

secondsInDay

```
public:static const double secondsInDay
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A constant value which represents a number of seconds in a day.

Current Date and Time

Use this member function to query the current time stamp.

currentTimeStamp

```
public:static ITimeStamp currentTimeStamp()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current timestamp.

Diagnostics

Use these member functions for diagnostic purposes.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ITimeStamp as an IString object which shows what the ITimeStamp represents in terms of date and time format.

OS/2 Considerations

To obtain locale-sensitive result, you must enable locale-based string operation by calling IString::enableInternationalization().

Windows Considerations

To obtain locale-sensitive result, you must enable locale-based string operation by calling IString::enableInternationalization().

asText

```
public:IText asText() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ITimeStamp as an IText object which shows what the ITimeStamp represents in terms of date and time format.

Manipulations

Use these operators to perform mathematical operations on timestamps.

operator +

```
public:ITimeStamp operator +(double seconds) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The given number of seconds is added to the ITimeStamp object, yielding a new ITimeStamp.

operator +=

```
public:ITimeStamp& operator +=(double seconds)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The ITimeStamp object is modified by adding the given number of seconds.

operator -

If the right-hand operand is a number of seconds, the ITimeStamp object subtracts the value, yielding a new ITimeStamp.

If the right-hand operand is an ITimeStamp object, the operator yields the number of seconds difference between the two timestamps. The difference is negative if the receiver ITimeStamp has timestamp value before the right-hand operand.

Overload 1

```
public:double operator -(const ITimeStamp& aTimeStamp) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITimeStamp operator -(double seconds) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -=

```
public:ITimeStamp& operator -=(double seconds)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The ITimeStamp object is modified by subtracting the given number of seconds.

Queries

Use this member function to query state information from the object.

asSeconds

```
public:double asSeconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of seconds from the reference date 01/01/2000 00:00:00.

Streaming Support

These members support streaming in and out ITimeStamp objects.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this object in from the passed data stream.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this object out to the passed data stream.

readFromStream

```
protected:void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Support function for streaming in time stamp objects from a stream.

writeToStream

```
protected:void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Support method for streaming time stamp objects out to a stream.

Type Conversions

Use these member functions to convert time stamps into legacy date or time objects.

operator IDate

```
public:operator IDate() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an IDate object. Only full days are counted, seconds that are a fraction of the day will be truncated.

operator ITime

```
public:operator ITime() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ITime object. Only the integral part of seconds that are a fraction of the day will be kept.

ITimeStamp - Associated Globals

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const ITimeStamp& aTimeStamp )
```

Streams this object out to an output stream, formatted as text.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const ITimeStamp& aTimeStamp )
```

Streams this object out to an output stream, formatted as text.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITimeStamp - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITypeRepresentation

This class is part of the implementation of IDataStream. Application code should make no direct use of it.

Class header file: ityperep.hpp

ITypeRepresentation - Hierarchy List

ITypeRepresentation

ITypeRepresentation - Member Functions and Data by Group

ITypeRepresentation - Inherited Member Functions and Data

Inherited Public Functions

ITypeRepresentation			
Definition	**Page Number**	**Definition**	**Page Number**
`operator !=`	316	`operator ==`	316

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IUnixPathNameParser

IUnixPathNameParser parses and formats Unix-style file system path names.

Class header file: ipathnam.hpp

IUnixPathNameParser - Hierarchy List

ICompoundNameParser

IUnixPathNameParser

IUnixPathNameParser - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct objects of this class.

~IUnixPathNameParser

```
public:virtual ~IUnixPathNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IUnixPathNameParser

Constructs an IUnixPathNameParser object.

Overload 1

```
public:IUnixPathNameParser(const IUnixPathNameParser& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a copy of the given object.

copyMe

The object to copy.

Overload 2

```
public:IUnixPathNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

operator =

```
public:IUnixPathNameParser&
  operator =( const IUnixPathNameParser& copyMe )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IUnixPathNameParser object to another.

copyMe

The object to copy.

Accessing the Separator

Use this member function to obtain the separator character used to separate path name components.

separator

```
public:virtual ioc::unichar_t separator() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the UniCode character that is used to separate path name components.

Parsing and Formatting

Use these member functions to parse and format path names.

formatName

```
public:virtual IHostPathName
  formatName( const ICompoundName& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts an ICompoundName into a platform-specific text string.

parseName

```
public:virtual ICompoundName parseName(const IHostPathName&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts a platform-specific text string into an ICompoundName object.

IUnixPathNameParser - Inherited Member Functions and Data

Inherited Public Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
~ICompoundNameParser	226	formatName	227
operator !=	226	operator ==	227
parseName	227	separator	226

Inherited Public Data

None

Inherited Protected Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
ICompoundNameParser	226	operator =	226
typeOf	227		

Inherited Protected Data

None

IVolume

This class provides a surrogate object for a file system volume.

Do not derive your own classes from this class.

Class header file: ifilesys.hpp

IVolume - Hierarchy List

IMStreamable
IFileSystemEntity
IDirectory
IVolume

IVolume - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy and destruct IVolume objects.

~IVolume

```
public:virtual ~IVolume()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IVolume

Constructs an IVolume object.

Overload 1

```
public:IVolume()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Overload 2

```
public:IVolume(const IVolume& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Class constants

Constants defined by this class.

kKind

```
public:static const IFileKind kKind
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

An IFileKind constant that identifies the IFile class. This constant is used for dynamic type checking.

Obtaining Volume Property Information

Use these member functions to obtain property information about a volume, such as the total amount of space, the amount of available space, whether it is local or remote, and whether it is removable.

freeSpace

```
public:IFileSize freeSpace() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the amount of available space on the volume, in bytes.

getRecord

```
public:IFSVolumeRecord* getRecord() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file system record of this object.

online

```
public:EState online() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether the volume is online.

Return

kTrue if the volume's contents are currently accessible, kFalse if not, and kUnkown if it could not be determined.

remote

```
public:EState remote() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether the volume is local or remote.

Return

kTrue if the volume is mounted remotely over a network, kFalse if it is local, and kUnkown if it could not be determined.

removable

```
public:EState removable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether the volume is a removable disk.

Return

kTrue if the volume is a removable disk such as a floppy or CD, kFalse if it is fixed, and kUnkown if it could not be determined.

totalSpace

```
public:IFileSize totalSpace() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the total size of the volume, in bytes.

Special Downcast Constructors

Use these constructors to create IVolume objects based on the given information, such as a host path name, a partial path name, a file system entity, or a file system volume record.

IVolume

Constructs an IVolume object.

Overload 1

```
public:IVolume(const IFileSystemEntity& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a file system entity object, with type checking.

copy If this entity refers to a physical volume, the constructed IVolume refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a volume.

Overload 2

```
public:IVolume(IFSVolumeRecord* Record)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IVolume object from a file system record.

Overload 3

```
public:IVolume( const IHostPathName& path,
         const ICompoundNameParser& = defaultParser ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IVolume object from a host-specific path name.

path A text string containing the host-specific path name.

parser The path name parser which should be used to parse the provided path. If no parser is specified, the default for the current system is used.

operator =

```
public:IVolume& operator =(const IFileSystemEntity& right)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns an IVolume to an entity object, with type checking.

right If this entity refers to a physical volume, the IVolume now refers to the same object. If not, an exception is thrown.

Exception

Exception	Description
IEntityTypeMismatch	Thrown if the specified entity is not a volume.

IVolume - Enumerations

EState

Return value for IVolume state access member functions.

kUnknown means that the state could not be determined, either because the host operating system does not support that type of state or because determining the state would be an unreasonably expensive operation.

IVolume - Inherited Member Functions and Data

Inherited Public Functions

IDirectory			
Definition	**Page Number**	**Definition**	**Page Number**
~IDirectory	228	createDirectory	229
createFile	229	createTemporary	230
current	232	deleteAllContents	230
getRecord	230	IDirectory	228
lookUp	230	operator =	228
setCurrent	232		

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
~IFileSystemEntity	281	connected	286
copyTo	284	defaultParser	283
deleteSelf	284	getRecord	282
hostPath	282	IFileSystemEntity	281
isA	283	modificationTime	282
moveTo	284	name	282
operator !=	286	operator =	282
operator ==	286	parent	285
path	283	readFromStream	285
setName	283	temporary	283
topmostParent	285	uniqueName	284
valid	286	volume	285
writeToStream	285		

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable writeToStream	200 202	readFromStream	201

IFileSystemEntity			
Definition	**Page Number**	**Definition**	**Page Number**
setInvalid	286	typeSafeAssign	283

Inherited Protected Data

None

IVolumeIterator

Iterates over over all mounted volumes on the system.

This class is designed to mimic a C++ pointer, similar to the iterator classes in the ANSI/ISO C++ draft standard. When the C++ Standard Library is available, IDirectoryIterator will inherit from input_iterator<IFileSystemEntity>.

Typical usage is as follows:

```
for (IVolumeIterator aVolume; aVolume; aVolume++)
{
    DoSomething( *aVolume );
}
```

Class header file: ifileitr.hpp

IVolumeIterator - Hierarchy List

IVolumeIterator

IVolumeIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct IVolumeIterator objects.

~IVolumeIterator

```
public:virtual ~IVolumeIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the IVolumeIterator object.

IVolumeIterator

Constructs an IVolumeIterator object.

Overload 1

```
public:IVolumeIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an empty IVolumeIteratorObject.

Overload 2

```
public:IVolumeIterator(const IVolumeIterator& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IVolumeIterator object that is copied from the given one.

Note that copying a volume iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

copy A constant reference to the object to be copied.

operator =

```
public:IVolumeIterator& operator =(const IVolumeIterator& copy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IVolumeIterator object to another.

Assigning a volume iterator does not do what you might expect. It creates another iterator object which refers to the same iteration. If you increment the new iterator, the original iterator is also incremented, because both objects actually refer to the same iteration.

Iterating Through Volumes

Use these operators to iterator through the volumes.

operator *

```
public:IVolume operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the IVolume that this iterator is iterating.

operator ++

Proceeds to the next volume in the iteration.

Overload 1

```
public:IVolumeIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IVolumeIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ->

```
public:IVolume* operator ->()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function allows the iterator to be used as if it were a normal C++ pointer, in expressions like:

```
anIterator->name()
```

Miscellaneous

Miscellaneous functions.

end

```
public:static const IVolumeIterator& end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an iterator that represents the end of iteration.

Testing and Comparing Volume Iterators

Use these member functions to compare or validate volume iterators.

operator !=

```
public:bool operator !=(const IVolumeIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests two volume iterators for inequality. They compare as equal if they are both past the end of their iteration, or if one was copy-constructed from the other.

operator ==

```
public:bool operator ==(const IVolumeIterator& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests this IVolumeIterator object against the given one for equality.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function is often used in the termination condition of a for or while loop. Returns false when there are no more volumes over which to iterate.

IVolumeIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IVolumeOffline

IVolumeOffline is an exception that indicates that an attempt has been made to operate on an offline or ejected volume.

Class header file: iexcfile.hpp

IVolumeOffline - Hierarchy List

IException

IDeviceError

IVolumeOffline

IVolumeOffline - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IVolumeOffline object from another.

~IVolumeOffline

Destructor.

```
public:virtual ~IVolumeOffline()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IVolumeOffline

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

Copy constructor.

```
public:IVolumeOffline(const IVolumeOffline&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

Constructor.

```
public:IVolumeOffline(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IVolumeOffline - Inherited Member Functions and Data

Inherited Public Functions

IDeviceError			
Definition	**Page Number**	**Definition**	**Page Number**
~IDeviceError	760	IDeviceError	760
nameW	761		

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IWin32PathNameParser

IWin32PathNameParser parses and formats Windows-style file system path names.

Class header file: ipathnam.hpp

IWin32PathNameParser - Hierarchy List

ICompoundNameParser

IWin32PathNameParser

IWin32PathNameParser - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, assign and destruct objects of this class.

~IWin32PathNameParser

```
public:virtual ~IWin32PathNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IWin32PathNameParser

Constructs an IWin32PathNameParser object.

Overload 1

```
public:IWin32PathNameParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Overload 2

```
public:IWin32PathNameParser(const IWin32PathNameParser& copyMe)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a copy of the given object.

copyMe

The object to copy.

operator =

```
public:IWin32PathNameParser&
  operator =( const IWin32PathNameParser& copyMe )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this assignment operator to assign one IWin32PathNameParser object to another.

copyMe

The object to copy.

Accessing the Separator

Use this member function to obtain the separator character used to separate path name components.

separator

```
public:virtual ioc::unichar_t separator() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Unicode character that is used to separate path name components.

Parsing and Formatting

Use these member functions to parse and format path names.

formatName

```
public:virtual IHostPathName
  formatName( const ICompoundName& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Convert an ICompoundName into a platform-specific text string.

parseName

```
public:virtual ICompoundName parseName(const IHostPathName&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts a platform-specific text string into an ICompoundName object.

IWin32PathNameParser - Inherited Member Functions and Data

Inherited Public Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
~ICompoundNameParser	226	formatName	227
operator !=	226	operator ==	227
parseName	227	separator	226

Inherited Public Data

None

Inherited Protected Functions

ICompoundNameParser			
Definition	**Page Number**	**Definition**	**Page Number**
ICompoundNameParser	226	operator =	226
typeOf	227		

Inherited Protected Data

None

Chapter 4. Text and International Classes

codecvt_base

The codecvt_base class defines constants indicating different codeset conversion results.

Note: This class is obsolete. The ANSI C++ Standard Library provides the same facility.

You should not derive from this class.

Class header file: icodecvt.hpp

codecvt_base - Hierarchy List

codecvt_base

codecvt_base - Member Functions and Data by Group

codecvt_base - Enumerations

result

Useful constants indicating different codeset conversion results.

ok Conversion completed successfully.

partial Not all source characters converted successfully. This can happen if the target buffer provides insufficient storage.

error Encountered a character that could not be converted.

noconv
No conversion was required.

codecvt_base - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

I0String

The I0String class provides arrays of characters. Objects of the I0String class are functionally equivalent to objects of the class IString with one major distinction: I0Strings are indexed starting at 0 instead of 1.

Note: A consequence of starting indexes at 0 is that you can no longer use the search functions as if they were boolean. For example:

```
a0String.indexOf( anotherString ) != a0String.includes( anotherString ).
```

© Copyright IBM Corp. 2001

You can freely intermix IStrings and I0Strings in a program. You can assign objects of one class values of the other type. You can pass objects of either class as parameters to functions requiring the other type.

Class header file: i0string.hpp

I0String - Hierarchy List

IString

I0String

I0String - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct I0String objects.

~I0String

```
public:~I0String()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an I0String object.

I0String

You can construct objects of this class in the following ways:

- Construct a NULL string.
- Construct a string with the ASCII representation of a given numeric value, supporting all variations of integer and double.
- Construct a string with a copy of the specified character data, supporting ASCII strings, characters, and IStrings. The character data passed is converted to its ASCII representation.
- Construct a string with contents that consist of copies of up to three buffers of arbitrary data (void*). Optionally, you only need to provide the length, in which case the I0String contents are initialized to a specified pad character. The default pad character is a blank.

These constructors can throw exceptions under the following conditions:

- Memory allocation errors

  Many factors dynamically allocate space and these allocation requests may fail. If so, the Open Class Library translates memory allocation errors into exceptions. Generally, such errors do not occur until you allocate an astronomical amount of storage.
- Out-of-range errors.

  These occur if you attempt to construct an I0String with a length greater than that supported by ulength_t.

Overload 1

```
public:I0String(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of a long numeric value.

Overload 2

```
public:I0String(unsigned anUnsigned)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of an unsigned numeric value.

Overload 3

```
public:I0String(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified IString.

Overload 4

```
public:I0String(unsigned short anUnsignedShort)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of an unsigned short numeric value.

Overload 5

```
public:I0String(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of a double numeric value.

Overload 6

```
public:I0String(char aChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the character. The string length is set to 1.

Overload 7

```
public:I0String(unsigned char anUnsignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the unsigned character. The string length is set to 1.

Overload 8

```
public:I0String(const char* pChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified ASCII string.

Overload 9

```
public:I0String(signed char aSignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the signed character. The string length is set to 1.

Overload 10

```
public:I0String(unsigned long anUnsignedLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of an unsigned long numeric value.

Overload 11

```
public:I0String(unsigned long long anUnsignedLongLong)
```

This is supported on **AIX OS/400 z/OS**

This constructor is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 12

```
public:I0String(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

This constructor is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 13

```
public:I0String(const unsigned char* pUnsignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified unsigned ASCII string.

Overload 14

```
public:I0String( const void* pBuffer1,
          ulength_t lenBuffer1,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 15

```
public:I0String( const void* pBuffer1,
          ulength_t lenBuffer1,
          const void* pBuffer2,
          ulength_t lenBuffer2,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 16

```
public:I0String( const void* pBuffer1,
          ulength_t lenBuffer1,
          const void* pBuffer2,
          ulength_t lenBuffer2,
          const void* pBuffer3,
          ulength_t lenBuffer3,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 17

```
public:I0String(const signed char* pSignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified signed ASCII string.

Overload 18

```
public:I0String(short aShort)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of a short numeric value.

Overload 19

```
public:I0String(int anInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with the ASCII representation of an integer numeric value.

Overload 20

```
public:I0String()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a NULL string.

Counting

Use these member functions to determine the number of the specified type.

occurrencesOf

Returns the number of occurrences of the specified IString, char*, char, or IStringTest. If you just want a boolean test, this is slower than I0String::indexOf.

Overload 1

```
public:ucount_t
  occurrencesOf( char aCharacter,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ucount_t
  occurrencesOf( const IStringTest& aTest,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ucount_t
  occurrencesOf( const IString& aString,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:ucount_t
  occurrencesOf( const char* pString,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Editing

These member functions are reimplemented to treat the position arguments as 0-based.

change

Changes occurrences of a specified pattern to a specified replacement string. You can specify the number of changes to perform. The default is to change all occurrences of the pattern. You can also specify the position in the receiver at which to begin.

The parameters are the following:

inputString
The pattern string as a reference to an object of type IString. The library searches for the pattern string within the receiver's data.

pInputString
The pattern string as a NULL-terminated string. The library searches for the pattern string within the receiver's data.

outputString
The replacement string as a reference to an object of type IString. It replaces the occurrences of the pattern string in the receiver's data.

pOutputString
The replacement string as a NULL-terminated string. It replaces the occurrences of the pattern string in the receiver's data.

startPos
The position to start the search at within the receiver's data. The default is 0.

numChanges
The number of patterns to search for and change. The default is I0String::kAllChanges, which causes changes to all occurrences of the pattern.

Overload 1

```
public:static I0String
  change( const IString& aString,
          const char* pInputString,
          const char* pOutputString,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:I0String&
  change( const char* pPattern,
          const IString& aReplacement,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:I0String&
  change( const IString& aPattern,
          const char* pReplacement,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static I0String
  change( const IString& aString,
          const char* pInputString,
          const IString& outputString,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:static I0String
  change( const IString& aString,
          const IString& inputString,
          const char* pOutputString,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:I0String&
  change( const IString& aPattern,
          const IString& aReplacement,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:static I0String
  change( const IString& aString,
          const IString& inputString,
          const IString& outputString,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:I0String&
  change( const char* pPattern,
          const char* pReplacement,
          uoffset_t startPos = 0,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

insert

Inserts the specified string at the specified location.

Overload 1

```
public:static I0String
  insert( const IString& aString,
          const IString& anInsert,
          uoffset_t index = IString::kEndOfString,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:I0String&
  insert( const char* pString,
          uoffset_t index = IString::kEndOfString,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:static I0String
  insert( const IString& aString,
          const char* pInsert,
          uoffset_t index = IString::kEndOfString,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:I0String&
  insert( const IString& aString,
          uoffset_t index = IString::kEndOfString,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

overlayWith

Replaces a specified portion of the receiver's contents with the specified string. The overlay starts in the receiver's data at the **index**, which defaults to 0. If **index** is beyond the end of the receiver's data, it is padded with the pad character, **padCharacter**.

Overload 1

```
public:static I0String
  overlayWith( const IString& aString,
               const IString& anOverlay,
               uoffset_t index = 0,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static I0String
  overlayWith( const IString& aString,
               const char* pOverlay,
               uoffset_t index = 0,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:I0String&
  overlayWith( const char* pString,
               uoffset_t index = 0,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:I0String&
  overlayWith( const IString& aString,
               uoffset_t index = 0,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

remove

Deletes the specified portion of the string (that is, the substring) from the receiver. You can use this function to truncate an I0String object at a specific position. For example:

```
aString.remove(8);
```

removes the substring beginning at index 8 and takes the rest of the string as a default.

Overload 1

```
public:static I0String
  remove( const IString& aString,
          uoffset_t startPos,
          ucount_t numChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:I0String& remove(uoffset_t startPos, ucount_t numChars)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:I0String& remove(uoffset_t startPos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static I0String
  remove( const IString& aString,
          uoffset_t startPos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Forward Searches

These member functions are reimplemented to treat the starting position of the search as a 0-based index.

indexOf

Returns the byte index of the first occurrence of the specified string within the receiver. If there are no occurrences, I0String::notFound is returned. In addition to IStrings, you can also specify a single character or an IStringTest.

Overload 1

```
public:uoffset_t
  indexOf( const IString& aString,
           uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOf( const char* pString,
           uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOf( const IStringTest& aTest,
           uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOf( char aCharacter,
           uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyBut

Returns the index of the first character of the receiver that is not in the specified set of characters. If there are no characters, I0String::notFound is returned. Alternatively, this function returns the index of the first character that fails the test prescribed by a specified IStringTest object.

Overload 1

```
public:uoffset_t
  indexOfAnyBut( const char* pValidChars,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOfAnyBut( const IStringTest& aTest,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOfAnyBut( const IString& aString,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOfAnyBut( char validChar,
                 uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyOf

Returns the index of the first character of the receiver that is a character in the specified set of characters. If there are no characters, I0String::notFound is returned. Alternatively, this function returns the index of the first character that passes the test prescribed by a specified IStringTest object.

Overload 1

```
public:uoffset_t
  indexOfAnyOf( const IStringTest& aTest,
                uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOfAnyOf( char searchChar,
                uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOfAnyOf( const IString& searchChars,
                uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOfAnyOf( const char* pSearchChars,
                uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfPhrase

```
public:uoffset_t
  indexOfPhrase( const IString& wordString,
                 ucount_t startWord = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the position of the first occurrence of the specified phrase in the receiver. If the phrase is not found, I0String::notFound is returned.

notFound

```
public:static const uoffset_t notFound
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You use this static constant in conjunction with the searching functions. It specifies the value searching functions return indicating the search failed.

Index Mapping

Use these member functions to convert arguments and results to the proper index base: 1 for arguments (because it relies on IString) and 0 for results (because it is 0-based itself).

adjustArg

Adjusts the specified index from 0- to 1-based.

Overload 1

```
protected:static offset_t adjustArg(offset_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static uoffset_t adjustArg(uoffset_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

adjustResult

Adjusts a function result from 1- to 0-based.

Overload 1

```
protected:static offset_t adjustResult(offset_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static uoffset_t adjustResult(uoffset_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Queries

These member functions are overridden to permit specification of the index as a 0-based value.

charType

```
public:IStringEnum::CharType charType(uoffset_t index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the type of the character at the specified index.

operator []

Returns a reference to the specified character of the string.

Note: If you call the non-const version of this function with an index beyond the end, the function extends the string.

Overload 1

```
public:const char& operator [](unsigned index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:char& operator [](signed long index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:char& operator [](signed index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:const char& operator [](signed long index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:const char& operator [](signed index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:char& operator [](unsigned long long index)
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 7

```
public:const char& operator [](unsigned long long index) const
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 8

```
public:char& operator [](signed long long index)
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 9

```
public:const char& operator [](signed long long index) const
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See ilanglvl.hpp.

Overload 10

```
public:const char& operator [](unsigned long index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 11

```
public:char& operator [](unsigned long index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 12

```
public:char& operator [](unsigned index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reverse Searches

These member functions permit searching the string in various ways. The lastIndexOf versions correspond to forward search indexOf members but start the search from the end of the string. These members return the index of the last character in the receiver I0String that satisfies the search criteria. Also, they accept an optional argument that specifies where the search is to begin. The default is to start searching at the end of the string. Searching proceeds from right to left for these members.

lastIndexOf

Returns the index of the last occurrence of the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The returned value is in the range 0 <= x <= **startPos** or I0String::notFound. The default of I0String::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, I0String::notFound is returned.

If you specify 0 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

Overload 1

```
public:uoffset_t
  lastIndexOf( const IString& aString,
               uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOf( const char* pString,
               uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOf( char aCharacter,
               uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOf( const IStringTest& aTest,
               uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyBut

Returns the index of the last character not in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of I0String::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, I0String::notFound is returned.

If you specify 0 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

Overload 1

```
public:uoffset_t
  lastIndexOfAnyBut( const IStringTest& aTest,
                     uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOfAnyBut( char validChar,
                     uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOfAnyBut( const char* pValidChars,
                     uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOfAnyBut( const IString& validChars,
                     uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyOf

Returns the index of the last character in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of I0String::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, I0String::notFound is returned.

If you specify 0 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

Overload 1

```
public:uoffset_t
  lastIndexOfAnyOf( const char* pSearchChars,
                    uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOfAnyOf( const IStringTest& aTest,
                    uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOfAnyOf( char searchChar,
                    uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOfAnyOf( const IString& searchChars,
                    uoffset_t endPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subset

Use these member functions when a subset of characters is required.

subString

Returns the specified portion of the string (that is, the substring) of the receiver.

The parameters are the following:

startPos
The starting position of the substring being extracted. If this position is beyond the end of the data in the receiver, this function returns a NULL I0String.

length The length of the substring to be extracted. If the length extends beyond the end of the receiver's data, the returned I0String is padded to the specified length with **padCharacter**. If you do not specify **length** and it defaults, this function uses the rest of the receiver's data starting from **startPos** for padding.

padCharacter
The character the function uses as padding if the requested length extends beyond the end of the receiver's data. The default **padCharacter** is a blank.

You can use this function to truncate an I0String object at a specific position. For example:

```
aString = aString.subString(0, 7);
```

returns the substring concluding with index 7 and discards the rest of the string.

Overload 1

```
public:I0String
  subString( uoffset_t startPos,
             ulength_t len,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:I0String subString(uoffset_t startPos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Word Operations

These member functions are reimplemented to treat the result index as 0-based.

indexOfWord

```
public:uoffset_t indexOfWord(ucount_t wordNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the index of the specified white-space-delimited word in the receiver. If the word is not found, I0String::notFound is returned.

I0String - Associated Globals

kEndOfString

```
static const uoffset_t kEndOfString
```

This static constant is used by the I0String::lastIndexOf, I0String::lastIndexOfAnyBut and I0String::lastIndexOfAnyOf member functions to define the starting position of the search. kEndOfString starts the search at the end of the string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

I0String - Inherited Member Functions and Data

Inherited Public Functions

IString			
Definition	**Page Number**	**Definition**	**Page Number**
~IString	475	asDebugInfo	482
asDouble	513	asInt	513
asLongLong	514	asString	482
asUnsigned	514	asUnsignedLongLong	514
b2c	479	b2d	479
b2x	479	c2b	480
c2d	480	c2x	481
center	482	change	482
copy	484	d2b	502
d2c	502	d2x	503
disableInternationalization	500	enableInternationalization	500
includes	503	includesDBCS	501
includesMBCS	501	includesSBCS	501
indexOf	493	indexOfAnyBut	494
indexOfAnyOf	494	insert	485
isAbbreviationFor	503	isAlphabetic	511
isAlphanumeric	511	isASCII	511
isBinaryDigits	511	isControl	512
isDBCS	501	isDigits	512
isGraphics	512	isHexDigits	512
isInternationalized	501	isLike	504
isLowerCase	512	isMBCS	501
isPrintable	512	isPunctuation	512
isSBCS	501	IString	475
isUpperCase	513	isValidDBCS	502
isValidMBCS	502	isWhiteSpace	513
lastIndexOf	507	lastIndexOfAnyBut	508
lastIndexOfAnyOf	509	leftJustify	485
length	505	lengthOfWord	515
lineFrom	509	lowerCase	486
numWords	516	occurrencesOf	481
operator &	497	operator &=	498
operator +	498	operator +=	498
operator <<=	510	operator =	499
operator >>=	510	operator ^	499
operator ^=	499	operator |	499
operator |=	500	operator ~	500
operator char *	514	operator signed char *	514
operator unsigned char *	515	overlayWith	486

IString			
Definition	**Page Number**	**Definition**	**Page Number**
remove	487	removeWords	516
reverse	487	rightJustify	488
size	506	space	488
strip	488	stripBlanks	489
stripLeading	490	stripLeadingBlanks	490
stripTrailing	491	stripTrailingBlanks	492
subString	510	translate	492
upperCase	493	word	516
wordIndexOfPhrase	516	words	517
x2b	495	x2c	496
x2d	496		

Inherited Public Data

IString			
Definition	**Page Number**	**Definition**	**Page Number**
kAllChanges	522	kEndOfString	522

I0String			
Definition	**Page Number**	**Definition**	**Page Number**
kEndOfString	346		

Inherited Protected Functions

IString			
Definition	**Page Number**	**Definition**	**Page Number**
applyBitOp	480	buffer	507
change	482	data	507
defaultBuffer	507	findPhrase	517
indexOfWord	495	initBuffer	496
insert	485	isAbbrevFor	513
isLike	504	lengthOf	507
nullBuf	507	occurrencesOf	481
overlayWith	486	readFromStream	510
setBuffer	497	strip	488
translate	492	writeToStream	510

Inherited Protected Data

IString			
Definition	**Page Number**	**Definition**	**Page Number**
maxLong zero	515 515	null	515

IBinaryFormat

IBinaryFormat provides functions to format and extract numbers in the binary number system. Binary numbers are strings of '0' and '1'. For example, the number 10_{10} can be represented by 1010_2.

$1010_2 := 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 0 * 2^0$

To format a number in the binary number system, you use the format member function:

```
IText bn = IBinaryFormat::format(10);
// bn = "1010"
```

To convert a binary string back into its numeric value. you can use the parseUnsignedLong() or the parseUnsignedLongLong() function. The latter is only supported if the compiler's long long support is turned on. By default, it is turned on. For example:

```
unsigned long i = IBinaryFormat::parseUnsignedLong("1010");
// i = 10
```

See also:

- IDecimalFormat
- IHexadecimalFormat

The following convenience functions convert a number string from one number system to another.

IText d2x(const IText& txt)
Convert a decimal string to a hexadecimal string.

IText x2d(const IText& txt)
Convert a hexadecimal string to a decimal string.

IText b2x(const IText& txt)
Convert a binary string to a hexadecimal string.

IText x2b(const IText& txt)
Convert a hexadecimal string to a binary string.

IText d2b(const IText& txt)
Convert a decimal string to a binary string.

IText b2d(const IText& txt)
Convert a binary string to a decimal string.

The following functions are only available if the compiler's long long support is turned on. By default, it is turned on.

IText d2xll(const IText& txt)
Convert a long long decimal string to a hexadecimal string.

IText x2dll(const IText& txt)
Convert a long long hexadecimal string to a decimal string.

IText b2xll(const IText& txt)
Convert a long long binary string to a hexadecimal string.

IText x2bll(const IText& txt)
Convert a long long hexadecimal string to a binary string.

IText d2bll(const IText& txt)
Convert a long long decimal string to a binary string.

IText b2dll(const IText& txt)
Convert a long long binary string to a decimal string.

Class header file: inumfrmt.hpp

IBinaryFormat - Hierarchy List

IBinaryFormat

IBinaryFormat - Member Functions and Data by Group

Formatting Binary Number

Use these member functions to format a number into a binary string.

format

Formats a number into a binary string.

Overload 1

```
public:static IText format(unsigned long n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IText format(unsigned long long n)
```

This is supported on **AIX OS/400 z/OS**

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Parsing Binary String

Use these member functions to parse a binary string an return its numeric value.

parseUnsignedLong

```
public:static unsigned long parseUnsignedLong(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a binary string and returns its numeric value. The argument should contain only valid binary symbols ('0's and '1's). Otherwise, results are undefined.

parseUnsignedLongLong

```
public:static unsigned long long
  parseUnsignedLongLong( const IText& txt )
```

This is supported on **AIX OS/400 z/OS**

Parses a binary string and returns its numeric value. The argument should contain only valid binary symbols ('0's and '1's). Otherwise, results are undefined.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

IBinaryFormat - Associated Globals

hexToBinary

```
static IText hexToBinary(const IText& hex)
```

Convert a hex string to a binary string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBinaryFormat - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IBitwiseCollation

IBitwiseCollation provides bitwise comparison services for text strings. IBitwiseCollation is a concrete class derived from ICollation. It provides the protocol for language-insensitive comparison of text strings. The comparison functions in this class perform a bitwise comparison of the numeric Unicode values in the text strings. ICollation::createCollation creates a bitwise comparison object when the POSIX locale ″C″ is given as the locale key.

Do not derive from this class.

Class header file: itxtcoll.hpp

IBitwiseCollation - Hierarchy List

ICollation

IBitwiseCollation

IBitwiseCollation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IBitwiseCollation objects.

~IBitwiseCollation

```
public:virtual ~IBitwiseCollation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IBitwiseCollation object.

IBitwiseCollation

Constructs an IBitwiseCollation object.

Overload 1

```
public:IBitwiseCollation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor for creating a bitwise comparison object.

Overload 2

```
protected:IBitwiseCollation(const IBitwiseCollation& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes an IBitwiseCollation object from another IBitwiseCollation object.

Assignment Operator

Use this operator to assign one IBitwiseCollation object to another.

operator =

```
protected:IBitwiseCollation&
  operator =( const IBitwiseCollation& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents with the IBitwiseCollation object being copied in.

Displayable names

Provides access to the displayable name.

displayName

```
public:virtual IText
  displayName( const ILocaleKey& desiredLanguage = ILocaleKey::defaultLocale ( ),
               ILocaleKey::ELookupStrategy strategy = ILocaleKey::kAnyLanguage ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the displayable name of a collation object in the specified language, using the given lookup strategy.

desiredLanguage
The locale which the current locale name of this collation object will be displayed in.

strategy
The look-up strategy for the display name of the current collation object. The default value is kAnyLanguage.

Exception

Exception	Description
IOutofSystemResource	If there is no display name for the given collation object.
IInvalidRequest	If the desired locale is supported but not installed.
IInvalidParameter	If the desired locale ID is invalid.

Implementation

Class implementation functions.

do_compare

```
protected:virtual int
  do_compare( const ioc::unichar_t* sourceLow,
              const ioc::unichar_t* sourceHigh,
              const ioc::unichar_t* targetLow,
              const ioc::unichar_t* targetHigh ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The actual comparison algorithm.

Returns 1 if the source is greater than the target, 0 if both are equal and -1 if the source is less than the target.

sourceLow
Source string beginning offset.

sourceHigh
Source string ending offset.

targetLow
Target string beginning offset.

targetHigh
Target string ending offset.

do_transform

```
protected:virtual IText do_transform(const IText& source) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The actual text transforming algorithm. Use IText::compare for the equality test of two transformed strings. Use transform when dealing with index building on a large amount of data as one example. Returns the transformed text by value.

source The source text to be transformed with.

IBitwiseCollation - Inherited Member Functions and Data

Inherited Public Functions

ICollation			
Definition	**Page Number**	**Definition**	**Page Number**
~ICollation	384	compare	384
createCollation	388	displayName	387
hash	385	isEqual	389
isGreaterThan	389	isLessThan	389
localeKey	387	setStrength	387
strength	388	transform	386

Inherited Public Data

None

Inherited Protected Functions

ICollation			
Definition	**Page Number**	**Definition**	**Page Number**
createUniCharChunk	384	do_compare	388
do_hash	386	do_transform	388
ICollation	384	operator =	384
setLocaleKey	388		

Inherited Protected Data

None

IBuffer

The IBuffer class defines the contents of an IString.

Class header file: ibuffer.hpp

IBuffer - Hierarchy List

IMRefCounted

IBuffer

IDBCSBuffer

IBuffer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IBuffer objects.

~IBuffer

```
protected:virtual ~IBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IBuffer object.

IBuffer

```
protected:IBuffer(IString::ulength_t newLen)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IBuffer object.

Allocation

Use these protected member functions to allocate and deallocate IBuffer objects.

allocate

```
protected:virtual IBuffer*
  allocate( IString::ulength_t bufLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a new buffer of the specified length.

operator delete

Deallocates a buffer.

Overload 1

```
protected:void
  operator delete( void* p,
                   const char* filename,
                   ioc::isize_t linenum )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:void operator delete(void* p)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator new

Allocates space for a buffer of the specified length. The returned pointer is an area the size of an IBuffer large enough to hold data of size **bufLen**.

Overload 1

```
protected:void*
  operator new( size_t t,
                const char* filename,
                ioc::isize_t linenum,
                IString::ulength_t bufLen )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:void* operator new(size_t t, IString::ulength_t bufLen)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Comparisons

Use these member functions to compare the IBuffer's contents to some other character array.

compare

```
public:virtual Comparison
  compare( const void* p,
           IString::ulength_t len ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the buffer's contents to the contents of the specified character array.

Diagnostics

Use these member functions to provide diagnostic information about the buffer.

asDebugInfo

```
public:virtual IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns information about the buffer's internal representation that you can use for debugging.

Editing

These member functions are called by the corresponding IString member functions to edit the buffer's contents.

center

```
public:virtual IBuffer*
  center( IString::ulength_t newLen,
          char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Centers the receiver within a string of the specified length.

change

```
public:virtual IBuffer*
  change( const char* pSource,
          IString::ulength_t sourceLen,
          const char* pTarget,
          IString::ulength_t targetLen,
          IString::uoffset_t startPos,
          IString::ucount_t numChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes occurrences of a specified pattern to a specified replacement string. You can specify the number of changes to perform. The default is to change all occurrences of the pattern. You can also specify the position in the receiver at which to begin.

The parameters are the following:

pSource
The pattern string as a NULL-terminated string. The library searches for the pattern string within the receiver's data.

sourceLen
The length of the source string.

pTarget
The target string as a NULL-terminated string. It replaces the occurrences of the pattern string in the receiver's data.

targetLen
The length of the target string.

startPos
The position to start the search at within the target's data.

numChanges
The number of patterns to search for and change.

copy

```
public:virtual IBuffer* copy(IString::ucount_t numCopies)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the receiver's contents with a specified number of replications of itself.

insert

```
public:virtual IBuffer*
  insert( const char* pInsert,
          IString::ulength_t insertLen,
          IString::uoffset_t pos,
          char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts the specified string after the specified location.

leftJustify

```
public:virtual IBuffer*
  leftJustify( IString::ulength_t newLen,
               char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Left-justifies the receiver in a string of the specified length. If the new length, **newLen**, is larger than the current length, the string is extended by the pad character **padCharacter**. The default pad character is a blank.

lowerCase

```
public:virtual IBuffer* lowerCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Translates all uppercase letters in the receiver to lowercase.

overlayWith

```
public:virtual IBuffer*
  overlayWith( const char* overlay,
               IString::ulength_t len,
               IString::uoffset_t pos,
               char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces a specified portion of the receiver's contents with the specified string. If **pos** is beyond the end of the receiver's data, it is padded with the pad character, **padCharacter**.

remove

```
public:virtual IBuffer*
  remove( IString::uoffset_t startPos,
          IString::ucount_t numChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes the specified portion of the string (that is, the substring) from the receiver. You can use this function to truncate an IString object at a specific position. For example:

```
aString.remove(8);
```

removes the substring beginning at index 8 and takes the rest of the string as a default.

reverse

```
public:virtual IBuffer* reverse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reverses the receiver's contents.

rightJustify

```
public:virtual IBuffer*
  rightJustify( IString::ulength_t newLen,
                char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Right-justifies the receiver in a string of the specified length. If the receiver's data is shorter than the requested length, **newLen**, it is padded on the left with the pad character, **padCharacter**. The default pad character is a blank.

strip

Strips both the leading and trailing character or characters. You can specify the character or characters as the following:

- A char* array
- An IStringTest object

The default is white space.

Overload 1

```
public:virtual IBuffer*
  strip( const IStringTest& aTest,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IBuffer*
  strip( const char* pChars,
         IString::ulength_t len,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

translate

```
public:virtual IBuffer*
  translate( const char* pInputChars,
             IString::ulength_t inputLen,
             const char* pOutputChars,
             IString::ulength_t outputLen,
             char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts all of the receiver's characters that are in the first specified string to the corresponding character in the second specified string.

upperCase

```
public:virtual IBuffer* upperCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Translates all lowercase letters in the receiver to uppercase.

Implementation

This member function helps implement this class.

initialize

```
protected:static IBuffer* initialize()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes the IBuffer object (sets up a NULL buffer, a DBCS table, and so forth).

NLS Testing

Corresponding IString member functions use these member functions to test the buffer's contents. These tests are character-set-specific.

includesDBCS

```
public:virtual bool includesDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are DBCS (double-byte character set), true is returned.

includesMBCS

```
public:virtual bool includesMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are MBCS (multiple-byte character set), true is returned.

includesSBCS

```
public:virtual bool includesSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are SBCS (single-byte character set), true is returned.

isDBCS

```
public:virtual bool isDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are DBCS, true is returned.

isMBCS

```
public:virtual bool isMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are MBCS, true is returned.

isSBCS

```
public:virtual bool isSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are SBCS, true is returned.

isValidDBCS

```
public:virtual bool isValidDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no DBCS characters have a 0 second byte, true is returned.

isValidMBCS

```
public:virtual bool isValidMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no MBCS characters have a 0 second byte, true is returned.

Protected Queries

This member function helps implement this class.

className

```
protected:virtual const char* className() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class (IBuffer).

Queries

Use these member functions to access various attributes of a buffer.

charType

```
public:virtual IStringEnum::CharType
  charType( IString::uoffset_t index ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the type of a character at the specified index.

contents

Returns the address of the buffer's contents.

Overload 1

```
public:const char* contents() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:char* contents()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

defaultBuffer

```
public:static IBuffer* defaultBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the address of the NULL buffer for the class. This is a static function.

fromContents

```
public:static IBuffer* fromContents(const char* pBuffer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the address of IBuffer using the specified pointer to its contents. This is a static function.

Note: It is important that **pBuffer** point to the actual beginning of data from an IBuffer object. The Open Class Library can return only values from the contents function of this class. Otherwise, if the returned IBuffer pointer is used, errors could occur.

length

```
public:IString::ulength_t length() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the length of the buffer's contents.

next

Returns a pointer to the next character, not the next byte, in the buffer.

Overload 1

```
public:virtual const char* next(const char* prev) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual char* next(const char* prev)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

null

```
public:IBuffer* null() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the address of the NULL buffer.

useCount

```
public:IString::ucount_t useCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of IStrings referring to the buffer.

Reallocation

Use these member functions to manage reallocation of IBuffers when strings' contents are modified.

checkAddition

```
public:static IString::ulength_t
  checkAddition( IString::ulength_t addend1,
                 IString::ulength_t addend2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Verifies that the two parameters, when added, do not overflow an unsigned value defined by IString::ulength_t.

checkMultiplication

```
public:static IString::ulength_t
  checkMultiplication( IString::ulength_t factor1,
                       IString::ulength_t factor2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Verifies that the two parameters, when multiplied, do not overflow an value defined by IString::ulength_t.

newBuffer

```
public:IBuffer*
  newBuffer( const void* p1,
             IString::ulength_t len1,
             const void* p2 = 0,
             IString::ulength_t len2 = 0,
             const void* p3 = 0,
             IString::ulength_t len3 = 0,
             char padChar = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allocates a new buffer and initializes it with the contents of up to three specified buffers.

The parameters are the following:

p1 The pointer to the first part to be copied into the data area of the new buffer. The first part is **len1** bytes long. If the pointer is NULL, the **padChar** is copied for **len1** bytes.

len1 The length, in bytes, of the first part to be copied into the new buffer.

p2 A pointer to the second part, immediately following the first part, to be copied into the data area of the new buffer. The second part is **len2** bytes long. If the pointer is NULL, the **padChar** is copied for **len2** bytes. If nothing is specified for **p2**, it is NULL.

len2 The length, in bytes, of the second part to be copied into the new buffer. If nothing is specified for **len2**, it defaults to 0 bytes.

p3 The pointer to the third part, immediately following the second part, to be copied into the data area of the new buffer. The third part is **len3** bytes long. If the pointer is NULL, the **padChar** is copied for **len3** bytes. If nothing is specified for **p3**, it is NULL.

len3 The length, in bytes, of the third part to be copied into the new buffer. If nothing is specified for **len3**, it defaults to 0 bytes.

padChar
The character to use as the pad in the cases of **p1**, **p2**, or **p3** being NULL. If you do not specify a **padChar**, it defaults to the character 0.

Note: If the sum of **len1**, **len2**, and **len3** is 0, a reference to the NULL buffer for this class is added and the address is returned.

overflow

```
public:static IString::ulength_t overflow()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Throws an exception when IBuffer::checkAddition or IBuffer::checkMultiplication detects an overflow.
Exception

Exception	Description
IInvalidRequest	An IBuffer request caused an overflow. Typically, this occurs during object construction or during an operation that grows an underlying IBuffer object. Likely culprits might be an IBuffer::newBuffer or IString::change call.

setDefaultBuffer

```
public:static void setDefaultBuffer(IBuffer* newDefaultBuffer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the default (NULL) buffer. The specified buffer must be comprised of a single NULL byte.

Search Initialization

These member functions help implement this class.

startBackwardsSearch

```
protected:virtual IString::uoffset_t
  startBackwardsSearch( IString::uoffset_t startPos,
                        IString::ulength_t searchLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes a search of type IString::lastIndexOf using the following criteria:

- If **searchLen** is greater than the length of the buffer, 0 is returned indicating an invalid search request.
- If the starting position is 0 or beyond the last **searchLen** bytes of the buffer, the position where the last **searchLen** bytes start in the buffer is returned.
- If the starting position is 1 through the last **searchLen** bytes, the value of **startingPos** is returned.

startSearch

```
protected:virtual IString::uoffset_t
  startSearch( IString::uoffset_t startPos,
               IString::ulength_t searchLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes a search of type IString::indexOf using the following criteria:

- If **startPos** is 0, the search uses a starting position of 1.
- If the specified **startPos** and **searchLen** result in an invalid search, 0 is returned. This usually occurs when the sum of **startPos** and **searchLen** is greater than the size of the buffer.

Searches

These member functions are called by the corresponding IString member functions to search the buffer's contents.

indexOf

Returns the byte index of the first occurrence of the specified string within the receiver. If there are no occurrences, 0 is returned.

Overload 1

```
public:virtual IString::uoffset_t
  indexOf( const char* pString,
           IString::ulength_t len,
           IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOf( const IStringTest& aTest,
           IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyBut

Returns the index of the first character of the receiver that is not in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that fails the test prescribed by a specified IStringTest object.

Overload 1

```
public:virtual IString::uoffset_t
  indexOfAnyBut( const IStringTest& aTest,
                 IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOfAnyBut( const char* pString,
                 IString::ulength_t len,
                 IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyOf

Returns the index of the first character of the receiver that is a character in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that passes the test prescribed by a specified IStringTest object.

Overload 1

```
public:virtual IString::uoffset_t
  indexOfAnyOf( const IStringTest& aTest,
                IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOfAnyOf( const char* pString,
                IString::ulength_t len,
                IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOf

Returns the index of the last occurrence of the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The returned value is in the range 0 <= x <= **startPos**. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOf( const IStringTest& aTest,
               IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOf( const char* pString,
               IString::ulength_t len,
               IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyBut

Returns the index of the last character not in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOfAnyBut( const IStringTest& aTest,
                     IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOfAnyBut( const char* pString,
                     IString::ulength_t len,
                     IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyOf

Returns the index of the last character in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOfAnyOf( const char* pString,
                    IString::ulength_t len,
                    IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOfAnyOf( const IStringTest& aTest,
                    IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subset

Use these member functions when a subset of characters is required.

subString

Returns a new IBuffer, of the same type as the previous one, containing the specified subset of characters.

The parameters are the following:

startPos
The index at which to start the substring. If **startPos** is 0, the function uses position 1. If **startPos** is beyond the end of the buffer, nothing is copied. The buffer is filled out by the specified padding character.

len The length to copy from the buffer. If the length extends beyond the end of the buffer, only the portion up to the end is copied. The buffer is then padded. If **len** is 0, a reference to the NULL buffer is returned.

padCharacter
Specifies the character the function uses to pad the copied string if less than **len** bytes have been copied from the source buffer.

Overload 1

```
public:virtual IBuffer*
  subString( IString::uoffset_t startPos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IBuffer*
  subString( IString::uoffset_t startPos,
             IString::ulength_t len,
             char padCharacter ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Testing

Corresponding IString member functions use these member functions to test the buffer's contents.

isAlphabetic

```
public:virtual bool isAlphabetic() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z','a'-'z'}, true is returned.

isAlphanumeric

```
public:virtual bool isAlphanumeric() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z','a'-'z','0'-'9'}, true is returned.

isASCII

```
public:virtual bool isASCII() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {0x00-0x7F}, true is returned.

isControl

```
public:virtual bool isControl() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are control characters.

Control characters are defined by the iscntrl() C Library function as defined in the cntrl locale source file and in the cntrl class of the LC_CTYPE category of the current locale. For example, on ASCII operating systems, control characters are those in the range {0x00-0x1F,0x7F}.

isDigits

```
public:virtual bool isDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9'}, true is returned.

isGraphics

```
public:virtual bool isGraphics() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are graphics characters.

Graphics characters are printable characters excluding the space character, as defined by the isgraph() C Library function in the graph locale source file and in the graph class of the LC_CTYPE category of the current locale. For example, on ASCII operating systems, graphics characters are those in the range {0x21-0x7E}.

isHexDigits

```
public:virtual bool isHexDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9','A'-'F','a'-'f'}, true is returned.

isLowerCase

```
public:virtual bool isLowerCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'a'-'z'}, true is returned.

isPrintable

```
public:virtual bool isPrintable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are printable characters.

Printable characters are defined by the isprint() C Library function as defined in the print locale source file and in the print class of the LC_CTYPE category of the current locale. For example, on ASCII systems, printable characters are those in the range {0x20-0x7E}.

isPunctuation

```
public:virtual bool isPunctuation() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If none of the characters is white space, a control character, or an alphanumeric character, true is returned.

isUpperCase

```
public:virtual bool isUpperCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z'}, true is returned.

isWhiteSpace

```
public:virtual bool isWhiteSpace() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are white-space characters.

White-space characters are defined by the isspace() C Library function as defined in the space locale source file and in the space class of the LC_CTYPE category of the current locale. For example, on ASCII systems, white-space characters are those in the range {0x09-0x0D,0x20}.

isDBCSLead

```
protected:static bool isDBCSLead(char inByte)
```

This is supported on **WIN OS/2**

If the input character is the first byte of a DBCS character, true is returned.

is32bytesAligned

```
protected:static bool is32bytesAligned
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBuffer - Type Definitions

Comparison

```
typedef enum { equal , greaterThan , lessThan } Comparison
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

These enumerators specify the possible valid return codes from IBuffer::compare:

equal The buffer's contents are equal to the contents of the specified character array.

greaterThan
The buffer's contents are greater than the contents of the specified character array.

lessThan
The buffer's contents are less than the contents of the specified character array.

IBuffer - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

ICCSId

The ICCSId class represents a **coded character set id** (CCSID) . A CCSID identifies the encoding scheme, character set, and code page, of data stored on various computer systems. A CCSID is a 2-byte binary identifier whose values range from 0 (X'0000') to 65535 (X'FFFF') .

For example on most AS/400 shipped in the U.S., the CCSID is 00037. This represents code page 00037, character set 00697, and encoding scheme 1100. An encoding scheme of 1100, represents EBCDIC, single-byte data. For more information on CCSIDs, character sets, encoding schemes, and code pages see your National Language Support book for your system.

Various other IOC classes can take a **code set** . There are two types of code sets within IOC, a portable code set and a system code set. On the AS/400 system, a system code set is the character form of a numeric CCSID. For example, CCSID 37, is equivalent to code set '37' . An AS/400 system code set is a character string with a maximum length of 5, each character of which represents a digit from 0 to 9.

The ICCSId class allows you to construct an object with a numeric CCSID or character code set. Regardless of how an ICCSId object is constructed the CCSID or code set can be obtained.

Class header file: iccsid.hpp

ICCSId - Hierarchy List

ICCSId

ICCSId - Member Functions and Data by Group

Constructors & Destructor

ICCSId

Constructs an ICCSId object.

Overload 1

```
public:ICCSId(const std::_CCSID_T& id)
```

This is supported on **OS/400**

This constructor is available if __IOC_ANSI_STREAM is defined.

Constructs an ICCSID from the class std::_CCSID_T which represents a numeric CCSID value.

Overload 2

```
public:ICCSId(const IText& codeset)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICCSId from a character representation of a CCSID.

Overload 3

```
public:explicit ICCSId(int id)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICCSId object given a numeric CCSID.

Overload 4

```
public:ICCSId(const _CCSID_T& id)
```

This is supported on **OS/400**

This constructor is available if __IOC_ANSI_STREAM is not defined.

Constructs an ICCSID from the class _CCSID_T which represents a numeric CCSID value.

Overload 5

```
public:ICCSId()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICCSId with an CCSID of 0.

Member Access

codeSet

```
public:IText codeSet() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character string (UCS-2) version of the CCSID.

isPortableCodeSet

```
public:bool isPortableCodeSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true, if the ICCSId represents an IOC portable code set. For a list of portable code sets supported by IOC refer to ICharacterSetIterator .

operator _CCSID_T

```
public:operator _CCSID_T() const
```

This is supported on **OS/400**

Converts an ICCSId object to an object of class _CCSID_T.

operator std::_CCSID_T

```
public:operator std::_CCSID_T() const
```

This is supported on **OS/400**

Converts an ICCSId object to an object of class std::_CCSID_T.

portableCodeSet

```
public:IText portableCodeSet() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the portable code set name as a character string (UCS-2) for the CCSID. If no portable code set exists for this CCSID a IText() is returned.

value

```
public:int value() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the numeric CCSID value.

ICCSId - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICharacterPropertyIterator

The ICharacterPropertyIterator class allows you to identify the set of Unicode characters with specific character properties.

Do not derive from this class.

Class header file: iunicode.hpp

ICharacterPropertyIterator - Hierarchy List

ICharacterPropertyIterator

ICharacterPropertyIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICharacterPropertyIterator objects.

~ICharacterPropertyIterator

```
public:virtual ~ICharacterPropertyIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICharacterPropertyIterator object.

ICharacterPropertyIterator

Constructs an ICharacterPropertyIterator object.

Overload 1

```
public:ICharacterPropertyIterator( const ICharacterPropertyIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ICharacterPropertyIterator object that is initialized by copying from the given one.

that The ICharacterPropertyIterator object being copied in.

Overload 2

```
public:ICharacterPropertyIterator( IUnicode::ECharacterProperty lowPropertyBound,
                                   IUnicode::ECharacterProperty highPropertyBound )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ICharacterPropertyIterator using the given values. Use this constructor to create an iterator that allows you to walk through a set of Unicode characters in the specified range.

lowPropertyBound
The low boundary of the character property range.

highPropertyBound
The high boundary of the character property range.

Assignment Operator

Use this operator to assign the given ICharacterPropertyIterator to the current one.
operator =

```
public:ICharacterPropertyIterator&
  operator =( const ICharacterPropertyIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents of *this iterator with the specified value.

that The ICharacterPropertyIterator object being copied in.

Getting the Current Unicode Character

Use this member function to obtain the current Unicode character in the range.
operator *

```
public:ioc::unichar_t operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current Unicode character in the range.

Incrementing the Iterator

Use these member functions to increment the character property iterator.
operator ++

Increments the iterator.

Overload 1

```
public:ICharacterPropertyIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function advances to the next Unicode character and returns the advanced character property iterator. When the end of range is detected, return *this.

Overload 2

```
public:ICharacterPropertyIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function advances to the next Unicode character and returns the advanced character property iterator. When the end of range is detected, returns *this.

int This int type represents the postfix form of ++ operator.

Resetting the Cursor Pointer

Use this member function to point the cursor at the first character property in the range.

reset

```
public:void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the cursor pointer to the first character property in the range.

Setting and Getting the Iterator Range

Use these member functions to get and set the character property iterator range.

range

```
public:void
  range( IUnicode::ECharacterProperty& lowPropertyBound,
         IUnicode::ECharacterProperty& highPropertyBound ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the character property iterator range.

lowPropertyBound
The low boundary of the character property range.

highPropertyBound
The high boundary of the character property range.

setRange

```
public:void
  setRange( IUnicode::ECharacterProperty lowPropertyBound,
            IUnicode::ECharacterProperty highPropertyBound )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the character property iterator range to a new range.

lowPropertyBound
The low boundary of the character property range.

highPropertyBound
The high boundary of the character property range.

Testing the Iterator Position

Use this member function to determine if the iterator is positioned at the end of the list.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Indicates that the iterator is at the end of the list.. Returns true if there are more elements on the list, otherwise returns false.

ICharacterPropertyIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICharacterReference

The ICharacterReference class is a simple proxy class for accessing characters safely. This class is used in IText and ITextIterator's APIs wherever a non-constant ioc::unichar_t& would otherwise be used. Changing a character through an ICharacterReference is safer and more efficient than changing a character through a ioc::unichar_t& because:

- Out-of-range character references are caught and throw an exception, rather than stomping on memory. (Although an ICharacterReference is not guaranteed to refer to the same character it did when created if the underlying IText is changed.)
- ICharacterReference knows about the storage-sharing mechanism and can protect against inadvertent changes to IText objects that share storage with the IText the user intends to change.
- ICharacterReference knows about the style mechanism and can protect against style propagation problems that may result from things like the caller getting rid of a paragraph delimiter.

ICharacterReference has no public constructors. The user must create an ICharacterReference using the [] operator or at() function on IText or the * or [] operator of one of the ITextIterator classes.

Do not derive from this class.

Class header file: itxtiter.hpp

ICharacterReference - Hierarchy List

ICharacterReference

ICharacterReference - Member Functions and Data by Group

Constructors & Destructor

Use the destructor in this group to destroy objects of class ICharacterReference.

~ICharacterReference

```
public:~ICharacterReference()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ICharacterReference object.

ICharacterReference

```
public:ICharacterReference(const ICharacterReference& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Assignment Operators

Use this operator to replace the character referred to in the current ICharacterReference object with the given one.

operator =

Changes the character referred to by the ICharacterReference.

Overload 1

```
public:ICharacterReference& operator =(ioc::unichar_t newChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ICharacterReference&
  operator =( const ICharacterReference& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes the character the ICharacterReference refers to to be the same as the character referred to by the that parameter. It does not change *this to refer to the same character as that. This overload exists to make expressions like the following legal:

```
someText[i] = someText[j]
```

that A constant ICharacterReference to the replacement character.

Miscellaneous Members

operator ioc::unichar_t

```
public:operator ioc::unichar_t() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obtaining Its Offset or a Pointer to the Text

Use these member functions to get a pointer to the text referred to or its offset.

offset

```
public:ioc::isize_t offset() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character offset within its IText of the character the ICharacterReference refers to.

Note: ICharacterReference is implemented as a single pointer to a ioc::unichar_t when possible, and as a pointer to an IText and a character offset the rest of the time. When an ICharacterReference holds a pointer to a character, this function returns (size_t)(-1) and the value returned by text() is not valid. (Consider this function to be obsolete.)

text

```
public:IText* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the IText the ICharacterReference refers to.

Note: ICharacterReference is implemented as a single pointer to a UCS2Char when possible, and as a pointer to an IText and a character offset the rest of the time. Therefore, the value returned by this function

may not be valid. To determine whether this value is valid, check the value returned by offset(). If it's (size_t)(-1), the value returned by text() is invalid.

Consider this function to be obsolete.

ICharacterReference - Inherited Member Functions and Data

Inherited Public Functions

ICharacterReference			
Definition	**Page Number**	**Definition**	**Page Number**
`unichar_t`	374		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICharacterSetIterator

ICharacterSetIterator provides APIs to iterate through the portable character code sets supported by the Transcoder Framework. ICharacterSetIterator does not find the code sets installed on your system.

For example, to search for a character set supported by the Transcoder Framework, you can do the following:

```
IText myCharSet("a candicate code set");
ICharacterSetIterator iter;
while (iter && *iter != myCharSet)
    iter++;
if (iter)
{
    ITranscoder* transcoder = ITranscoder::createTranscoder(*iter);
}
```

This class provides APIs that are compatible with ANSI C++ standard library.

You should not derive from this class.

Class header file: itrancod.hpp

ICharacterSetIterator - Hierarchy List

ICharacterSetIterator

ICharacterSetIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICharacterSetIterator objects.

~ICharacterSetIterator

```
public:virtual ~ICharacterSetIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICharacterSetIterator object.

ICharacterSetIterator

Constructs an ICharacterSetIterator object.

Overload 1

```
public:ICharacterSetIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor for ICharacterSetIterator.

Overload 2

```
public:ICharacterSetIterator(const ICharacterSetIterator& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor for ICharacterSetIterator.

source The ICharacterSetIterator object copied from.

Advancing the Iterator

Use these member functions to advance the character code set iterator to the next character set.

operator ++

Advances to the next character set.

Note: If this function takes no parameters, it serves as a prefix increment operator++. For example, you can do the following:

```
while (iter && *iter != myCharSet) ++iter;
```

Overload 1

```
public:ICharacterSetIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances to the next character set. This returns a pointer to the next character set.

Overload 2

```
public:ICharacterSetIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances to the next character set. The int parameter is not used by the client. This returns a pointer to the next character set.

Assignment Operator

Use this operator to replace the current ICharacterSetIterator with the given one.

operator =

```
public:ICharacterSetIterator&
  operator =( const ICharacterSetIterator& right )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

right The right-hand side ICharacterSetIterator to be assigned to the current one.

Comparing ICharacterSetIterator Objects

Use these member functions to compare the given ICharacterSetIterator object against the current one for equality.

operator !=

```
public:bool operator !=(const ICharacterSetIterator& right) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines the inequality of the two code set iterators. This function compares the portable names of the character sets pointed by this iterator and another iterator.

Returns true if the portable name of the character sets pointed to by this iterator and another iterator are not the same; otherwise, returns false.

right The given code set iterator to compare with.

operator ==

```
public:bool operator ==(const ICharacterSetIterator& right) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines the equality of the two code set iterators. This function compares the portable names of the character sets pointed by this iterator and another iterator.

Returns true if the portable name of the character sets pointed to by this iterator and another iterator are the same; otherwise, returns false.

right The given code set iterator to compare with.

Identifying the Character Set

Use this member funtion to get the portable name of the current character set pointed to by the iterator.

operator *

```
public:const IText operator *()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the portable name of the current character set pointed to by the iterator. Clients can use (*iter) with the ITranscoder::createTranscoder factory function to create a transcoder. See an example in the class description.

Resetting the Iterator

Use this member function to reset the iterator to point to the first character set.

reset

```
public:void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets this code set iterator pointing to its first character set.

Testing for the Iteration End

Use this member function to determine if the iterator has reached the end.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function detects if the iterator has reached the end of the iteration. Returns true if the iterator reaches the end; otherwise returns false.

ICharacterSetIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICharacterStyle

The ICharacterStyle class is an abstract base class for all character-based styles.

Do not derive your own classes from this class.

Class header file: itxtstyl.hpp

ICharacterStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextFixedPitchStyle
ITextBoldfaceStyle
ITextUnderlineStyle
ITextOutlineStyle
ITextColorStyle
ITextItalicStyle
ITextStrikethroughStyle
ITextUneditableStyle
ITextTypefaceStyle
ITextBitmapStyle
ITextPointSizeStyle
ITextSuperSubscriptStyle

ICharacterStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICharacterStyle objects.

~ICharacterStyle

```
public:virtual ~ICharacterStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ICharacterStyle object.

ICharacterStyle

Constructs an ICharacterStyle object.

Overload 1

```
protected:ICharacterStyle(const ICharacterStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor. Use this constructor to create an ICharacterStyle object that is copied from the given one. ICharacterStyle is a constant reference to the ICharacterStyle that is copied.

Overload 2

```
protected:ICharacterStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor. Use this constructor to create an empty ICharacterStyle object.

Assignment Operator

Use this operator to assign the given ICharacterStyle object to the target one.

operator =

```
protected:ICharacterStyle& operator =(const ICharacterStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Obtaining the Propagation Type

Use these member functions to get the style's propagation type.

propagates

```
public:virtual ITextStyle::EStylePropagation propagates() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the style's propagation type. For ICharacterStyle, this is ITextStype::kPropagateByCharacter.

ICharacterStyle - Inherited Member Functions and Data

Inherited Public Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

Inherited Protected Data

None

ICnrDate

The ICnrDate structure is used to create date column objects for the IContainerControl.

Class header file: idate.hpp

ICnrDate - Hierarchy List

ICnrDate

ICnrDate - Member Functions and Data by Group

Structure Definition

ICnrDate structure definition.

day

```
public:unsigned char day
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This field represents the Day.

month

```
public:unsigned char month
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This field represents the Month.

year

```
public:unsigned short year
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This field represents the Year.

ICnrDate - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICnrTime

The ICnrTime structure is used to create time column objects for the IContainerControl.

Class header file: itime.hpp

ICnrTime - Hierarchy List

ICnrTime

ICnrTime - Member Functions and Data by Group

Structure Definition

ICnrTime structure definition.

hours

```
public:unsigned char hours
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Hours.

minutes

```
public:unsigned char minutes
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Minutes.

seconds

```
public:unsigned char seconds
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Seconds.

ucReserved

```
public:unsigned char ucReserved
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reserved.

ICnrTime - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICollation

ICollation is an abstract class which provides Unicode text comparison services. ICollation provides language-sensitive text comparison when you want to compare text data according to the alphabetical ordering rules of a natural language rather than the numeric values of the character encoding set. Language-sensitive processing is required for case-insensitive ordering. For example, without using the language-sensitive processing, in the Roman script the letter Z would be ordered before the letter a.

Language-sensitive text comparisons are based on text collation objects that provide correct collation and selection for a particular natural language. End users specify a preferred text collation object in their locale. Text comparison functions use the preferred object to analyze text according to the rules of that language. The host- specific collation services use Win32 APIs on Windows. For AIX, OS/2 and OS400, the POSIX APIs are used.

The host-specific collation class is derived from ICollation which provides the protocol for text-ordering. The client can create the collation object based on the current locale using the locale key, ILocaleKey, and the level of comparison. A locale key is equivalent to the cross-environment locale item identifier and the comparison level defines how the text will be ordered in a locale. The comparison level (or collation strength) are primary, secondary and tertiary. The strength defines what will be ignored in a language sensitive text comparison.

An ICollation instance can be created by calling ICollation::createCollation with a locale key and the comparison level: A locale key is used to identify the current locale, which can be obtained by creating an ILocaleKey instance. For a locale key, if it is a valid host locale key, the host-specific collation object will be created. If a locale key is not a valid host locale key, an exception will be thrown. The clients can define how the text will be ordered in a locale by specifying the level of comparison. In general, clients tend to compare the strings at the same comparison level in an application. However, the collation strength can be set to a different comparison level. The default strength is kTertiaryDifference.

IBitwiseCollation can be directly constructed or called through ICollation::createCollation with the POSIX ″C″ locale key. ICollation also provides a set of functions that do simple ordering tests and return a boolean value. These functions just call the compare function directly:

- isGreaterThan
- isEqual
- isLessThan

The following example compares the primary differences between text1 and text2 strings in the french locale:

```
// create an ILocaleKey object (French)
ILocaleKey frLoc(IText("FR_FR"));
ICollation*  order =
    ICollation::createCollation(frLoc, ICollation::kPrimaryDifference);
int result = order->compare(text1, text2);
// do something
```

For this release, it is not recommended that you derive from ICollation since there is no registration services yet with createCollation(). Open Class Library provides the derived class IBitwiseCollation.

Class header file: itxtcoll.hpp

ICollation - Hierarchy List

ICollation

IPortableCollation

IBitwiseCollation

ICollation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICollation objects.

~ICollation

```
public:virtual ~ICollation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICollation object.

ICollation

Constructs an ICollation object.

Overload 1

```
protected:ICollation(const ICollation& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes an ICollation object from another ICollation object.

Overload 2

```
protected:ICollation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor for creating a collation object.

Assignment Operator

Use this operator to assign one collation object to another.

operator =

```
protected:ICollation& operator =(const ICollation& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents with the ICollation object being copied in.

Comparison/Transforming/Hashing

Compares, hashes, and transforms text based on the collation.

compare

Compares character data stored in two different text objects.

Returns information about whether a string is less than, greater than or equal to another string. The compare function that takes Unicode pointers

as text range is merely for the compatibility with the ANSI C++ standard collate class. For substring text ordering capability, use ITextIterator.

Overload 1

```
public:int
  compare( ITextIterator sourceBegin,
           ITextIterator sourceEnd,
           ITextIterator targetBegin,
           ITextIterator targetEnd ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns 1 if the source is greater than the target, 0 if both are equal and -1 if the source is less than the target.

sourceBegin
Iterator whose current offset is at the beginning of the source string.

sourceEnd
Iterator whose current offset is at the end of the source string.

targetBegin
Iterator whose current offset is at the beginning of the target string.

targetEnd
Iterator whose current offset is at the end of the target string.

Exception

Exception	Description
IInvalidParameter	If sourceEnd is less than sourceBegin, or targetEnd is less than targetBegin.

Overload 2

```
public:int
  compare( const ioc::unichar_t* sourceLow,
           const ioc::unichar_t* sourceHigh,
           const ioc::unichar_t* targetLow,
           const ioc::unichar_t* targetHigh ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:int compare(const IText& source, const IText& target) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns 1 if the source is greater than the target, 0 if both are equal and -1 if the source is less than the target.

source The source text to be compared with.

target The text string that is to be compared to the source text.

hash

Creates a long integer that is unique for the source string. If two objects are identical, their hash values should be the same. This is useful for dictionary hashing.

Overload 1

```
public:ioc::numeric_t hash(const IText& source) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ioc::numeric_t
  hash( const ioc::unichar_t* sourceLow,
        const ioc::unichar_t* sourceHigh ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ioc::numeric_t
  hash( ITextIterator sourceBegin,
        ITextIterator sourceEnd ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

transform

Creates an IText value that is compared lexicographically to the original text.

For any two strings, the results of doing bitwise comparison on the transformed strings will be the same as calling compare() on the original strings. Use IText::compare for the equality test of two transformed strings. Use transform when dealing with index building on a large amount of data.

Overload 1

```
public:IText transform(const IText& source) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the transformed text by value.

source The source text to be transformed with.

Overload 2

```
public:IText
  transform( ITextIterator sourceBegin,
             ITextIterator sourceEnd ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the transformed text by value.

sourceBegin
Iterator whose current offset is at the beginning of the source string.

sourceEnd
Iterator whose current offset is at the end of the source string.

Exception

Exception	Description
IInvalidParameter	If sourceEnd is less than sourceBegin.

do_hash

```
protected:virtual ioc::numeric_t
  do_hash( const ioc::unichar_t* sourceLow,
           const ioc::unichar_t* sourceHigh ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The actual hashing implementation.

Returns the hash value based on the source string's Unicode values.

sourceLow
Source string beginning offset.

sourceHigh
Source string ending offset.

Displayable Name

Provides access to the displayable name.

displayName

```
public:virtual IText
  displayName( const ILocaleKey& desiredLanguage = ILocaleKey::defaultLocale ( ),
               ILocaleKey::ELookupStrategy strategy = ILocaleKey::kAnyLanguage ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the displayable name of a collation object.

Specify the desired locale and look-up strategy for which the display name is based on. Refer to the ILocaleKey class description for details on look-up strategy.

desiredLanguage
The locale which the current locale name of this collation object will be displayed in.

strategy
The look-up strategy for the display name of the current collation object. The default value is kAnyLanguage.

Exception

Exception	Description
IOutofSystemResource	If there is no display name for the given collation object.
IInvalidRequest	If the desired locale is supported but not installed.
IInvalidParameter	If the desired locale ID is invalid.

Getter/Setter for Information Access

Use these member functions to get and set member data of an ICollation object.

localeKey

```
public:ILocaleKey localeKey() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the locale key of the collation object.

setStrength

```
public:void setStrength(ECollationStrength newStrength)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the new strength, the new comparison level.

strength

```
public:ECollationStrength strength() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the comparison level of the collation object.

Implementation Methods

Class implementation methods.

do_compare

```
protected:virtual int
  do_compare( const ioc::unichar_t* sourceLow,
              const ioc::unichar_t* sourceHigh,
              const ioc::unichar_t* targetLow,
              const ioc::unichar_t* targetHigh ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The actual comparison algorithm. Returns 1 if the source is greater than the target, 0 if both are equal and -1 if the source is less than the target.

sourceLow
Source string beginning offset.

sourceHigh
Source string ending offset.

targetLow
Target string beginning offset.

targetHigh
Target string ending offset.

do_transform

```
protected:virtual IText do_transform(const IText& source) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The actual text transforming algorithm.

Use IText::compare for the equality test of two transformed strings. Use transform when dealing with index building on a large amount of data as one example. Returns the transformed text by value.

source The source text to be transformed with.

setLocaleKey

```
protected:void setLocaleKey(const ILocaleKey& newLocale)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the locale key of the collation object to a new locale.

Instantiation Helper

Factory method used to create new ICollation derived objects.

createCollation

```
public:static ICollation*
  createCollation( const ILocaleKey& localeId = ILocaleKey::defaultLocale ( ),
                   ECollationStrength strength = kTertiaryDifference )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

An ICollation object can only be created by calling this static member.

Returns a pointer to the constructed ICollation instance.

localeId
Locale identifier. The default value for this argument is the current locale.

strength
The collation strength. The default value is kTertiaryDifference.

Exception

Exception	Description
IInvalidRequestion	If no collation object can be found.

Simple Comparison Helper Functions

Use these member functions to compare text based on the collation.

isEqual

```
public:virtual bool
  isEqual( const IText& sourceText,
           const IText& targetText ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Simple ordering test function. Tests the linguistic equality of two strings.

Returns true if the result of calling compare returns 0. Otherwise, returns false. It is equivalent to compare(source, target) == 0.

sourceText
The source text to be compared with.

targetText
The text string that is to be compared to the source text.

isGreaterThan

```
public:virtual bool
  isGreaterThan( const IText& sourceText,
                 const IText& targetText ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Simple ordering test function. Tests the linguistic equality of two strings.

Returns true if the result of calling compare returns 1. Otherwise, returns false. It is equivalent to compare(source, target) == 1.

sourceText
The source text to be compared with.

targetText
The text string that is to be compared to the source text.

isLessThan

```
public:virtual bool
  isLessThan( const IText& sourceText,
              const IText& targetText ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Simple ordering test function. Tests the linguistic equality of two strings.

Returns true if the result of calling compare returns -1. Otherwise, returns false. It is equivalent to compare(source, target) == -1.

sourceText
The source text to be compared with.

targetText
The text string that is to be compared to the source text.

ICollation - Enumerations

ECollationStrength

ECollationStrength is used to specify the comparison level.

Each Unicode character is assigned ordering priority: primary, secondary, tertiary, and no differences, as follows:

- Base letters represent a primary difference.
- Diacritical marks on the same base letter represent a secondary difference.
- Uppercase and lowercase versions of the same base letter represent a tertiary difference.
- Two characters are considered ″no different″ when they have equivalent Unicode spellings. Typically this appears in cases with expanding and contracting characters in which multiple characters combine to create a single ordering difference.

Note: For POSIX, there is no support for comparing strings with different comparison level. Therefore, the comparison result of two strings in primary, secondary or tertiary strength will be the same. For example, ″abc″ is always compared greater than ″ABC″ in primary, secondary, or tertiary strength.

ETextComparisonResult

Identifies the result of Unicode text strings comparison.

```
kSourceLess; source text < target.
kSourceEqual; source text == target.
kSourceGreater; source text > target.
```

ICollation - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

ICollation			
Definition	**Page Number**	**Definition**	**Page Number**
createUniCharChunk	384		

Inherited Protected Data

None

ICollationIterator

ICollationIterator provides the protocol for iterating through host collation objects. Create an ICollationIterator instance to step through all available collation services.

Class header file: icolitr.hpp

ICollationIterator - Hierarchy List

ICollationIterator

ICollationIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICollationIterator objects.

~ICollationIterator

```
public:~ICollationIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ICollationIterator object.

ICollationIterator

Constructs an ICollationIterator object.

Overload 1

```
public:ICollationIterator( ECollationType type = kHostCollationsOnly )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor for creating an ICollationIterator object.

type The collation object type. The default value is kHostCollationsOnly.

Overload 2

```
public:ICollationIterator(const ICollationIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor for the ICollationIterator object. Initializes an ICollationIterator object from another ICollationIterator object.

Assignment Operator

Use this operator to assign one ICollationIterator object to another.

operator =

```
public:ICollationIterator&
  operator =( const ICollationIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents of *this with the specified value.

that The ICollationIterator object being copied in.

Comparison Operators

Use these operators to compare the given ICollationIterator object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ICollationIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if two collation iterators are not the same. Returns true if the two collation iterators are not the same, false otherwise.

operator ==

```
public:bool operator ==(const ICollationIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if two collation iterators are the same. Returns true if the two collation iterators are the same, false otherwise.

Cursor actions

Use these member functions to control the iterator's cursor position.

create

```
public:ICollation*
  create( ICollation::ECollationStrength strength = ICollation::kTertiaryDifference ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the current collation object in the list. Returns the created collation object by value.

strength
The comparision level. The default value is kTertiaryDifference.

localePOSIXID

```
public:IText localePOSIXID() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the portable name of the locale(s) of the selected collation. Each collation may be referred to by more than one locale key or no locale. For example, US_EN and US_UK both use "American English" collation for text ordering. For a portable collation object "Foo", there may be no corresponding locale(s).

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if the iterator had consumed the last list item. Returns true if there are no more collation object to iterate through; otherwise, returns false.

reset

```
public:void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the cursor pointer to the first item on the list.

Displayable Name

Use this member function to get the displayable name.

displayName

```
public:IText
  displayName( const ILocaleKey& desiredLanguage = ILocaleKey::defaultLocale ( ),
               ILocaleKey::ELookupStrategy strategy = ILocaleKey::kAnyLanguage ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the displayable name for the collation object.

Fills in the displayable name for the current collation table in the desired language. The ILocaleKey class description contains more information on the display name versus key name of a locale. Returns the displayable name of the current collation table by value.

desiredLanguage
The desired language for the current collation object name. The default value is to use the current locale.

strategy
The look-up strategy for the display name of the current collation object. The default value is kAnyLanguage.

Exception

Exception	Description
IOutofSystemResource	If there is no display name for the given collation.
IInvalidRequest	If the desired locale is supported but not installed.
IInvalidParameter	If the desired locale ID is invalid.

Increment operators

Use these member functions to increment the collation iterator.

operator ++

Advances to the next collation table.

Overload 1

```
public:ICollationIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Increments the iterator, before the operation is performed (using the prefix increment operator). The int type represents the postfix form of the ++ operator.

Exception

Exception	Description
IInvalidRequest	If the last collation object has been consumed and advancing request is made by the clients.

Overload 2

```
public:ICollationIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Increments the iterator, after the operation is performed.
Exception

Exception	Description
IInvalidRequest	If the last collation object has been consumed and advancing request is made by the clients.

ICollationIterator - Enumerations

ECollationType

The type of collation objects to iterate through. Currently, only host-specific collation objects are supported.

kHostCollationsOnly
List the installed host collation objects only.

ICollationIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IConstTextIterator

The IConstTextIterator class provides a forward iterator for a const IText object. It provides read-only sequential access to the characters in an IText, and can be used as a pointer to an individual character in an IText.

This class conforms to the protocol for an STL-compatible random-access iterator.

Do not derive from this class.

Class header file: itxtiter.hpp

IConstTextIterator - Hierarchy List

IConstTextIterator

IConstTextIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IConstTextIterator objects.

~IConstTextIterator

```
public:~IConstTextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the IConstTextIterator object.

IConstTextIterator

Constructs an IConstTextIterator object.

Overload 1

```
public:IConstTextIterator(const IConstTextIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IConstTextIterator that is copied from the given one.

Overload 2

```
public:IConstTextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IConstTextIterator object that leaves the iterator in an undefined state. You must call the assignment operator before you can use a default-constructed iterator.

Assignment Operator

Use this operator to replace the current IConstTextIterator with the given one.

operator =

```
public:IConstTextIterator&
  operator =( const IConstTextIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Points the current IConstTextIterator to the same character position in the same IText pointed to by the given one specified as the that parameter.

Comparing IConstTextIterator Objects

Use these operators to compare the given IConstTextIterator object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the iterators refer to different character positions in the same IText.

that The IConstTextIterator object to be compared with this one.

Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

operator ==

```
public:bool operator ==(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if both iterators refer to the same character position in the same IText.

that The IConstTextIterator object to be compared with this one.
Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

Copying an ITextIterator Object

Use this member function to copy the current IConstTextIterator into a created ITextIterator object.

operator ITextIterator

```
public:operator ITextIterator() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ITextIterator that refers to the same character in the same IText as the character referred to by this, the current, ITextIterator.

Determining the Character the Iterator Refers To

Use the operators and member functions in this group to identify the character or character position the iterator refers to.

operator *

```
public:ioc::unichar_t operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the character the iterator refers to.
Exception

Exception	Description
IInvalidParameter	If the iterator refers to an out-of-range character position on its IText.

operator <

```
public:bool operator <(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the current IConstTextIterator object refers to a character with a lower offset than the given one does.

that An IConstTextIterator object containing the character to be tested.
Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

operator <=

```
public:bool operator <=(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the current IConstTextIterator object refers to the same character as the given one does, or to one with a lower offset.

that An IConstTextIterator object containing the character to be tested.
Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

operator >

```
public:bool operator >(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the current IConstTextIterator object refers to a character with a higher offset than the given one does.

that An IConstTextIterator object containing the character to be tested.
Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

operator >=

```
public:bool operator >=(const IConstTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the current IConstTextIterator object refers to the same character as the given one does, or to one with a higher offset.

that An IConstTextIterator object containing the character to be tested.
Exception

Exception	Description
IInvalidParameter	If both iterators do not refer to the same IText object.

operator []

```
public:ioc::unichar_t operator [](ioc::numeric_t offset) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character that is the specified number of offsets, or positions, away from the one the iterator refers to. For example, if iter points to

position 2 in a function, iter[3] will return the character at position 5 in that function, not the character at position 3. The iterator itself is left unchanged by this operation.

offset The number of offsets.

Identifying the IText Object of the Iterator

Use this member function to identify the IText object to which the iterator belongs and acts on.

text

```
public:const IText* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the IText object the iterator refers to.

Incrementing or Decrementing the Iterator

Use these operators to move the iterator, traversing characters.

operator ++

Advances the iterator to the next character position, that is, the position with the next highest offset. This function performs no error checking; rather, it depends on operator*() for error checking.

Overload 1

```
public:IConstTextIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the iterator before it was advanced.

Overload 2

```
public:IConstTextIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to this.

operator +=

```
public:IConstTextIterator& operator +=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator the number of character positions specified by distance. It advances the iterator toward the end of the text, that is, toward higher character offsets. This function performs no error checking; it depends on operator*() for error checking. Returns a reference to this.

distance
The number of character positions to advance the text.

operator -

```
public:IConstTextIterator
  operator -( ioc::numeric_t distance ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator --

Advances the iterator to the previous character position, that is, the one with the next-lower offset. This routine performs no error checking; it depends on operator*() for error checking.

Overload 1

```
public:IConstTextIterator& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to this.

Overload 2

```
public:IConstTextIterator operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the iterator before it was advanced.

operator -=

```
public:IConstTextIterator& operator -=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator by the distance of character positions toward the beginning of the text, that is, toward lower character offsets. Returns a reference to this. This routine performs no error checking; it depends on operator*() for error checking.

distance
The number of character positions to advance the text.

Obtaining Information About the Characters Referred To

Use these member functions to obtain the number of characters in the range marked by the iterators or to obtain the offset of the character referred to.

offset

```
public:ioc::isize_t offset() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the offset within the IText of the character the iterator refers to.

operator -

```
public:ioc::numeric_t
  operator -( const IConstTextIterator& that ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of character positions between the two iterators. This has the same effect as offset() - that.offset().

that The given iterator used with this one to mark the text range.

Exception

Exception	Description
IInvalidParameter	If the iterators refer to different IText objects.

IConstTextIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IDate

IDate represents specified dates. It returns language-sensitive information, such as names of days and months, in the language defined by the user's system. It also provides general day-handling and date-handling functions. Externally, dates are composed of :

- A year
- A month within that year
- A day within that month

Internally, dates are stored a true Julian date, i.e. the number of days from January 1, 4713 B.C, taking into account the Gregorian reforms. The day offset is stored as an unsigned long value, so it can represent dates up to approximately 11,700,000 year A.D.

The Open Class Library also lets you specify the day within the year.

Class header file: idate.hpp

IDate - Hierarchy List

IDate

IDate - Member Functions and Data by Group

Constructors & Destructor

You can construct objects of this class in the following ways:

- Use the default constructor, which returns the current day.
- Give the year, month, and day for the desired day. These parameters can be in either month/day/year or day/month/year order.
- Give the year and day of the year for the desired day.
- Use IDate::today to return the current date.
- Copy another IDate object.
- Give the Julian day number, as a long.
- Give a container details CDATE structure.

IDate

Constructs a date object from a number of different types of date specifications.

Overload 1

```
public:IDate(int aYear, int aDay)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IDate from the year and day of the year. The day of year is the number of days starting at January 1.

aYear The year to construct from. The passed day must be a legal day within this year.
aDay The day within the year to construct from.

Overload 2

```
public:IDate(Month aMonth, int aDay, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to pass parameters in month/day/year order.
aYear The year to construct from. The passed day must be a legal day within this year and month.
aMonth
The month within the passed year to construct from.
aDay The day within the month to construct from.

Overload 3

```
public:IDate(const IDate& aDate)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IDate by copying another IDate object.
aDate The source IDate object to copy construct from.

Overload 4

```
public:IDate(ioc::unumeric_t julianDayNumber)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IDate(const ICnrDate& cnrDate)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IDate from a container details ICnrDate structure.
cnrDate
The ICnrDate object to construct from.

Overload 6

```
public:IDate(int aDay, Month aMonth, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to construct a date object from parameters specified in day/month/year order.
aYear The year to construct from. The passed day must be a legal day within this year and month.
aMonth
The month within the passed year to construct from.
aDay The day within the month to construct from.

Overload 7

```
public:IDate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a date object containing the current day.

Comparison Operators

Use these member functions to compare two IDates. Use any of the full complement of comparison operators and apply the natural meaning.

operator !=

```
public:bool operator !=(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another date object for inequality.

aDate The other date object to compare to this one for inequality.

Return

true if this object is not equal to the passed object, else false.

operator <

```
public:bool operator <(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another date object for relative magnitude.

aDate The date to compare to this object for relative magnitude.

Return

true if this object represents a date prior to the passed date.

operator <=

```
public:bool operator <=(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another date object for relative magnitude or equality.

aDate The date to compare to this object for relative magnitude or equality.

Return

true if this object represents a date prior to or equal to the passed date.

operator ==

```
public:bool operator ==(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another object for equality.

aDate The other date object to compare to this one for equality.

Return

true if this object is equal to the passed date object, else false.

operator >

```
public:bool operator >(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another date object for relative magnitude.

aDate The date to compare to this object for relative magnitude.

Return

true if this object represents a date subsequent to the passed date.

operator >=

```
public:bool operator >=(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this object to another date object for relative magnitude or equality.

aDate The date to compare to this object for relative magnitude or equality.

Return

true if this object represents a date subsequent to or equal to the passed date.

Conversion Members

Use these member functions to retrieve other representations of the date.

asICnrDate

```
public:ICnrDate asICnrDate() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a container ICnrDate structure for this date.

julianDate

```
public:ioc::unumeric_t julianDate() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Julian day number of this object. The function uses the true definition of a Julian date, which means it returns the number of days from January 1, 4713 B.C.

Current Date

Use this member function when you need the current date.

today

```
public:static IDate today()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the current date. This static function can be used as an IDate constructor.

Day Queries

Use these member functions to access the day portion of an IDate object.

dayName

Returns the name of the day of week, either of this object's date or of a passed day of week enum.

Overload 1

```
public:static IString dayName(DayOfWeek aDay)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

aDay The day of week whose formatted name to get.

Overload 2

```
public:IString dayName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

dayNameW

Returns the name of the day of week, either of this object's date or of a passed day of week enum. It is functionally equivalent to its IString version.

Overload 1

```
public:static IText dayNameW(DayOfWeek aDay)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText dayNameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

dayOfMonth

```
public:int dayOfMonth() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the day in the receiver's month as an integer from 1 to 31.

dayOfWeek

```
public:DayOfWeek dayOfWeek() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the index of this object's day of the week: Monday through Sunday.

dayOfYear

```
public:int dayOfYear() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the day in the receiver's year as an integer from 1 to 366.

Diagnostics

These members provide an IString representation for an IDate object and have the capability to output the object to a stream. The formats include both mm-dd-yy and strftime conversion specifications. Often, you use these members to write trace information when debugging.

asString

Returns the IDate as an IString. By default, the string is formatted with the format string associated with the current locale. For compatibility reason, a two-digit year is always used.

Note: In previous releases of the Open Class Library, a two-digit year was used by default.

Overload 1

```
public:IString asString(YearFormat yearFmt = yy) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to format the date using one of a set of standard formats represented by the YearFormat enum.

yearFmt

Specifies how the system should display the year. If you do not specify the format, the default is yyyy. Use the enumeration IDate::YearFormat for valid values.

OS/2 Considerations

To obtain locale-sensitive result, you must enable locale-based string operation by calling IString::enableInternationalization().

Windows Considerations

To obtain locale-sensitive result, you must enable locale-based string operation by calling IString::enableInternationalization().

Overload 2

```
public:IString asString(const char* fmt) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this member function to format the date using a string of formatting parameters. These parameters give you total control over the fields to be formatted and where they are placed.

fmt Gives the conversion specifier, which is a character string you can use to describe how to format the date. Use the date specifiers that are valid in the C function strftime. The conversion specifiers that apply to IDate and their meanings are listed in the following table.

ITime::asString provides the conversion specifiers that apply to ITime. For more information about the strftime function, refer to the VisualAge for C++: C Library Reference:

Specifier	Meaning
%a	Insert abbreviated weekday name of locale.
%A	Insert full weekday name of locale.
%b	Insert abbreviated month name of locale.
%B	Insert full month name of locale.
%c	Insert date and time of locale.
%d	Insert day of the month (01-31).
%j	Insert day of the year (001-366).
%m	Insert month (01-12).
%U	Insert week number of the year (00-53) where Sunday is the first day of the week.
%w	Insert weekday (0-6) where Sunday is 0.
%W	Insert week number of the year (00-53) where Monday is the first day of the week.
%x	Insert date representation of locale.
%y	Insert year without the century (00-99).
%Y	Insert year with the century (e.g. 1998).

asText

Returns the IDate as an IText. By default, the string is formatted with the format string associated with the current locale. A four-digit year is always used. Please refer to their corresponding asString() functions for more details.

Overload 1

```
public:IText asText(const ioc::unichar_t* fmt) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText asText(YearFormat yearFmt = yyyy) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

General Date Queries

These member are static. They provide general IDate utilities independent of specific IDates. Typically, you use them to determine calendar information or to convert IDate enumeration data to string values.

daysInMonth

```
public:static int daysInMonth(Month aMonth, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of days in a specified month of a specified year. You must specify the year in yyyy format, that is, not relative to 1900.

aMonth
The month whose days to get.

aYear The year within which the month falls. Specify this value in yyyy format, that is, not relative to 1900.

daysInYear

```
public:static int daysInYear(int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of days in a specified year. You must specify the year in in yyyy format, that is, not relative to 1900.

aYear The year whose days to get. You must specify this value in yyyy format, that is, not relative to 1900.

Implementation

These members initialize an IDate object.

initialize

```
protected:IDate& initialize(Month aMonth, int aDay, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function calculates the Julian day number and sets this object's date to that Julian date.

aYear The year to initialize from. The passed day must be a legal day within this year and month.

aMonth
The month within the passed year to initialize from.

aDay The day within the month to initialize from.

Manipulation

Use these member functions to update an IDate object using addition or subtraction of another IDate object. Use any of the full complement of addition or subtraction operators and apply the natural meaning.

operator +

```
public:IDate operator +(int numDays) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function adds an integral number of days to the left-hand operand, yielding a new IDate which is returned by value.

numDays

The number of days to add to this object to create the return value.

operator +=

```
public:IDate& operator +=(int numDays)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function adds the given number of days to this date object.

numDays

An integer specifying the number of days to add to this date object.

operator -

This function subtracts an integral number of days from the left-hand operand, yielding a new IDate. If the right-hand operand is also an IDate, the operator returns the number of days between the dates. If the right-hand operand is a number of days, then the operator returns a new date object representing the resulting date.

Overload 1

```
public:IDate operator -(int numDays) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

numDays

The number of days to subtract from this date to create the return value.

Overload 2

```
public:ioc::numeric_t operator -(const IDate& aDate) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

aDate The date to subtract from this object.

operator -=

```
public:IDate& operator -=(int numDays)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function subtracts the given number of days from this date object.

numDays

An integer specifying the number of days to subtract from this date object.

Month Queries

Use these member functions to access the month portion of an IDate object.

monthName

Returns the name of the month of either this object's month, or of a passed month enumeration value.

Overload 1

```
public:static IString monthName(Month aMonth)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

aMonth
The month whose name to format.

Overload 2

```
public:IString monthName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

monthNameW

Returns the name of the month of either this object's month, or of a passed month enumeration value. It is functionally equivalent to its IString version.

Overload 1

```
public:static IText monthNameW(Month aMonth)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText monthNameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

monthOfYear

```
public:Month monthOfYear() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the index of this object's month of the year: January through December.

Streaming Operators

Use these operators to stream IData objects.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function streams this object in from the passed data stream. Any existing content is overwritten by the streamed in information.

fromWhere
The stream to stream this object in from.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function streams this date object out to the passed data stream.

toWhere
The stream to stream this object out to.

Validation

Use these static members to validate the passed-date data. They test the validity of a given day and provide a leap year test for a given year.

isLeapYear

```
public:static bool isLeapYear(int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns a boolean indicating whether the specified year is a leap year. You must specify the year in yyyy format, that is, not relative to 1900.

aYear The year to test for leap year. You must specify this value in yyyy form, that is, not relative to 1900.

Return

true if the passed year is a leap year, else false.

isValid

This function indicates whether the specified date is valid. You must specify the year in yyyy format, that is, not relative to 1900.

Overload 1

```
public:static bool isValid(int aYear, int aDay)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to determine if the date is valid for a date that you pass in as a year and day.

aYear The year to test for validity. You must specify this value in yyyy form, that is, not relative to 1900.

aDay The day within the year to test for validity.

Return

true if the passed year and day are valid, else false.

Overload 2

```
public:static bool isValid(int aDay, Month aMonth, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to determine if the date is valid for a date that you pass in as a day, month, and year.

aYear The year to test for validity. You must specify this value in yyyy form, that is, not relative to 1900.

aMonth

The month within the passed year to test for validity.

aDay The day within the month to test for validity.

Return

true if the passed day, month, and year are valid, else false.

Overload 3

```
public:static bool isValid(Month aMonth, int aDay, int aYear)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to determine whether a date is valid for a date you pass in as a month, a day, and a year.

aYear The year to test for validity. You must specify this value in yyyy form, that is not relative to 1900.

aMonth

The month within the passed year to test for validity.

aDay The day within the month to test for validity.

Return

true if the passed month, day, and year are valid, else false.

Year Queries

Use this member function to access the year portion of an IDate object.

year

```
public:int year() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the receiver's year. The returned value is in the yyyy format.

IDate - Type Definitions

DayOfWeek

```
typedef enum { Monday = 0 , Tuesday , Wednesday , Thursday , Friday ,
  Saturday , Sunday } DayOfWeek
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A typedef that provides the values Monday through Sunday for the days of the week.

Month

```
typedef enum { January = 1 , February , March , April , May , June ,
  July , August , September , October , November , December } Month
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A typedef that provides the values January through December for the months of the year.

YearFormat

```
typedef enum { yy , yyyy } YearFormat
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A typedef that specifies the number of digits in the year for the default asString format (yy or yyyy).

IDate - Associated Globals

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IDate& aDate )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IDate& aDate )
```

Streams this object out to an output stream, formatted as text.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDate - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IDate			
Definition	**Page Number**	**Definition**	**Page Number**
`readFromStream`	400	`writeToStream`	400

Inherited Protected Data

None

IDBCSBuffer

The IDBCSBuffer class implements the version of IString contents that supports mixed double-byte character set (DBCS) characters. This class also supports UNIX multiple-byte character set (MBCS) characters. This class ensures that multiple-byte characters are processed properly.

The use of this class is transparent to the user of class IString.

Class header file: idbcsbuf.hpp

IDBCSBuffer - Hierarchy List

IMRefCounted

IBuffer

IDBCSBuffer

IDBCSBuffer - Member Functions and Data by Group

Constructors & Destructor

The constructor for this class is protected. Only IDBCSBuffer::allocate and IBuffer::initialize can call the constructor.

~IDBCSBuffer

```
protected:~IDBCSBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IDBCSBuffer object.

IDBCSBuffer

```
protected:IDBCSBuffer(IString::ulength_t bufLength)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IDBCSBuffer object.

Allocation

Use these member functions to reimplement the allocation member functions as public.

allocate

```
public:virtual IDBCSBuffer*
  allocate( IString::ulength_t newLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a new buffer of the specified length.

Comparisons

Use these member functions to compare the IDBCSBuffer's contents to some other character array.

compare

```
public:virtual Comparison
  compare( const void* p,
           IString::ulength_t len ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the buffer's contents to the contents of the specified character array.

Editing

Use these member functions to reimplement the following IString versions of IBuffer members. The following member functions are called by the corresponding IString members to edit the buffer's contents.

center

```
public:virtual IDBCSBuffer*
  center( IString::ulength_t newLen,
          char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Centers the receiver within a string of the specified length.

change

```
public:virtual IDBCSBuffer*
  change( const char* pSource,
          IString::ulength_t sourceLen,
          const char* pTarget,
          IString::ulength_t targetLen,
          IString::uoffset_t startPos,
          IString::ucount_t numChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes occurrences of a specified pattern to a specified replacement string. You can specify the number of changes to perform. The default is to change all occurrences of the pattern. You can also specify the position in the receiver at which to begin.

The parameters are the following:

pSource
The pattern string as a NULL-terminated string. The library searches for the pattern string within the receiver's data.

sourceLen
The length of the source string.

pTarget
The target string as a NULL-terminated string. It replaces the occurrences of the pattern string in the receiver's data.

targetLen
The length of the target string.

startPos
The position to start the search at within the target's data.

numChanges
The number of patterns to search for and change.

insert

```
public:virtual IDBCSBuffer*
  insert( const char* pInsert,
          IString::ulength_t insertLen,
          IString::uoffset_t pos,
          char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts the specified string after the specified location.

leftJustify

```
public:virtual IDBCSBuffer*
  leftJustify( IString::ulength_t newLen,
               char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Left-justifies the receiver in a string of the specified length. If the new length, **newLen**, is larger than the current length, the string is extended by the pad character, **padCharacter**. The default pad character is a blank.

lowerCase

```
public:virtual IDBCSBuffer* lowerCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Translates all uppercase letters in the receiver to lowercase.

overlayWith

```
public:virtual IDBCSBuffer*
  overlayWith( const char* overlay,
               IString::ulength_t len,
               IString::uoffset_t pos,
               char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces a specified portion of the receiver's contents with the specified string. If **pos** is beyond the end of the receiver's data, it is padded with the pad character (**padCharacter**).

remove

```
public:virtual IDBCSBuffer*
  remove( IString::uoffset_t startPos,
          IString::ucount_t numChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes the specified portion of the string (that is, the substring) from the receiver. You can use this function to truncate an IString object at a specific position. For example:

```
aString.remove(8);
```

removes the substring beginning at index 8 and takes the rest of the string as a default.

reverse

```
public:virtual IDBCSBuffer* reverse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reverses the receiver's contents.

rightJustify

```
public:virtual IDBCSBuffer*
  rightJustify( IString::ulength_t newLen,
                char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Right-justifies the receiver in a string of the specified length. If the receiver's data is shorter than the requested length, **newLen**, it is padded on the left with the pad character, **padCharacter**. The default pad character is a blank.

strip

Strips both the leading and trailing character or characters. You can specify the character or characters as the following:

- A char* array
- An IStringTest object

The default is white space.

Overload 1

```
public:virtual IDBCSBuffer*
  strip( const IStringTest& aTest,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IDBCSBuffer*
  strip( const char* pChars,
         IString::ulength_t len,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

translate

```
public:virtual IDBCSBuffer*
  translate( const char* pInputChars,
             IString::ulength_t inputLen,
             const char* pOutputChars,
             IString::ulength_t outputLen,
             char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts all of the receiver's characters that are in the first specified string to the corresponding character in the second specified string.

upperCase

```
public:virtual IDBCSBuffer* upperCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Translates all lowercase letters in the receiver to uppercase.

NLS Testing

Use these member functions to reimplement the following IString versions of IBuffer members. The corresponding IString members use these members to test the buffer's contents. These tests are character-set-specific.

includesDBCS

```
public:virtual bool includesDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are DBCS (double-byte character set), true is returned.

includesMBCS

```
public:virtual bool includesMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are MBCS (multiple-byte character set), true is returned.

includesSBCS

```
public:virtual bool includesSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are SBCS (single-byte character set), true is returned.

isDBCS

```
public:virtual bool isDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are DBCS, true is returned.

isMBCS

```
public:virtual bool isMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are MBCS, true is returned.

isSBCS

```
public:virtual bool isSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are SBCS, true is returned.

isValidDBCS

```
public:virtual bool isValidDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no DBCS characters have a 0 second byte, true is returned.

isValidMBCS

```
public:virtual bool isValidMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no MBCS characters have a 0 second byte, true is returned.

Protected Queries

These member functions help implement this class.

charLength

Returns the number of bytes in the character whose first byte is pointed to by char *. This is a static function.

Overload 1

```
protected:static ioc::isize_t charLength(char const* pChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:ioc::isize_t charLength(IString::uoffset_t pos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:static ioc::isize_t
  charLength( char const* pChar,
              mbstate_t* pMBState )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
protected:ioc::isize_t
  charLength( IString::uoffset_t pos,
              mbstate_t* pMBState ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

className

```
protected:const char* className() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class (IDBCSBuffer).

maxCharLength

```
protected:static ioc::isize_t maxCharLength()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum number of bytes in a multiple-byte character. This is a static function.

prevCharLength

Returns the number of bytes in the preceding character to the one at the specified offset.

Overload 1

```
protected:ioc::isize_t
  prevCharLength( IString::uoffset_t pos,
                  mbstate_t* pMBState ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:ioc::isize_t prevCharLength(IString::uoffset_t pos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Testing

These member functions help implement this class.

isCharValid

```
protected:bool
  isCharValid( IString::uoffset_t pos,
               const char* pValidChars,
               IString::ucount_t numValidChars ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the character at the specified index is in the set of valid characters, true is returned.

The parameters are the following:

pos The position in the receiver's buffer for the validity check.

Note: It is important that this position not be the second byte of a DBCS character. If it is, you might get false results.

pValidChars
The string of the valid characters. It can contain a mixture of DBCS and SBCS characters.

numValidChars
The size of this string of valid characters.

isDBCS1

```
protected:bool isDBCS1(IString::uoffset_t pos) const
```

This is supported on **OS/2**

If the byte at the specified offset is the first byte of DBCS, true is returned.

Note: The Open Class Library provides this function only for compatibility with prior library versions. We recommend using IDBCSBuffer::charLength to determine if the byte is part of a multiple-byte character.

isPrevDBCS

```
protected:bool isPrevDBCS(IString::uoffset_t pos) const
```

This is supported on **OS/2**

If the preceding character to the one at the specified offset is a DBCS character, true is returned.

Note: The Open Class Library provides this function only for compatibility with prior library versions. We recommend using IDBCSBuffer::prevCharLength to determine if the preceding byte is part of a multiple-byte character.

isSBC

If the byte pointed to by the specified character is a single-byte character, true is returned. This is a static function.

Overload 1

```
protected:static bool isSBC(char const* pChar, mbstate_t* pMBState)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static bool isSBC(char const* pCh
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Queries

Use these member functions to reimplement the following IString versions of IBuffer members.

charType

```
public:virtual IStringEnum::CharType
  charType( IString::uoffset_t index ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the type of a character at the specified index.

next

Returns a pointer to the next character, not the next byte, in the buffer.

Overload 1

```
public:virtual const char* next(const char* prev) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual char* next(const char* prev)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Search Initialization

These member functions initialize search data.

startBackwardsSearch

```
protected:virtual IString::uoffset_t
  startBackwardsSearch( IString::uoffset_t startPos,
                        IString::ulength_t searchLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes a search of type IString::lastIndexOf following these criteria:

- If **searchLen** is greater than the length of the buffer, 0 is returned indicating an invalid search request.
- If the starting position is 0 or beyond the last **searchLen** bytes of the buffer, the position where the last **searchLen** bytes start in the buffer is returned.
- If the starting position is 1 through the last **searchLen** bytes, the value of **startingPos** is returned.

startSearch

```
protected:virtual IString::uoffset_t
  startSearch( IString::uoffset_t startPos,
               IString::ulength_t searchLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes a search of type IString::indexOf following these criteria:

- If **startPos** is 0, the search uses a starting position of 1.
- If the specified **startPos** and **searchLen** result in an invalid search, 0 is returned. This usually occurs when the sum of **startPos** and **searchLen** is greater than the size of the buffer.

Searches

Use these member functions to reimplement the following IString versions of IBuffer search members.

indexOf

Returns the byte index of the first occurrence of the specified string within the receiver. If there are no occurrences, 0 is returned.

Overload 1

```
public:virtual IString::uoffset_t
  indexOf( const IStringTest& aTest,
           IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOf( const char* pString,
           IString::ulength_t len,
           IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyBut

Returns the index of the first character of the receiver that is not in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that fails the test prescribed by a specified IStringTest object.

Overload 1

```
public:virtual IString::uoffset_t
  indexOfAnyBut( const char* pString,
                 IString::ulength_t len,
                 IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOfAnyBut( const IStringTest& aTest,
                 IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyOf

Returns the index of the first character of the receiver that is a character in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that passes the test prescribed by a specified IStringTest object.

Overload 1

```
public:virtual IString::uoffset_t
  indexOfAnyOf( const char* pString,
                IString::ulength_t len,
                IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  indexOfAnyOf( const IStringTest& aTest,
                IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOf

Returns the index of the last occurrence of the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The returned value is in the range 0 <= x <= **startPos**. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 or 1 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOf( const char* pString,
               IString::ulength_t len,
               IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOf( const IStringTest& aTest,
               IString::uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyBut

Returns the index of the last character not in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOfAnyBut( const IStringTest& aTest,
                     IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOfAnyBut( const char* pString,
                     IString::ulength_t len,
                     IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyOf

Returns the index of the last character in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of 0 starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 0 or 1 for **startPos**, this function returns 0 indicating the search target was not found.

Overload 1

```
public:virtual IString::uoffset_t
  lastIndexOfAnyOf( const IStringTest& aTest,
                    IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IString::uoffset_t
  lastIndexOfAnyOf( const char* pString,
                    IString::ulength_t len,
                    IString::uoffset_t startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subset

Use these member functions to reimplement the following IString versions of IBuffer subsetting members.

subString

Returns a new IBuffer, of the same type as the previous one, containing the specified subset of characters.

The parameters are the following:

startPos
The index at which to start the substring. If **startPos** is 0, the function uses position 1. If **startPos** is beyond the end of the buffer, nothing is copied. The buffer is filled out by the specified padding character.

len The length to copy from the buffer. If the length extends beyond the end of the buffer, only the portion up to the end is copied. The buffer is then padded. If **len** is 0, a reference to the NULL buffer is returned.

padCharacter
Specifies the character the function uses to pad the copied string if less than **len** bytes have been copied from the source buffer.

Overload 1

```
public:virtual IDBCSBuffer*
  subString( IString::uoffset_t startPos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual IDBCSBuffer*
  subString( IString::uoffset_t startPos,
             IString::ulength_t len,
             char padCharacter ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Testing

Corresponding IString member functions use these member functions to test the buffer's contents.

Note: In the Windows environment, the Testing functions are implemented using the wide char functions of the C runtime library.

On other platforms, IDBCSBuffer inherits the testing functions from the IBuffer class.

isAlphabetic

```
public:virtual bool isAlphabetic() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z','a'-'z'}, true is returned.

isAlphanumeric

```
public:virtual bool isAlphanumeric() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all characters are in {'A'-'Z','a'-'z','0'-'9'}, true is returned.

isASCII

```
public:virtual bool isASCII() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {0x00-0x7F}, true is returned.

isControl

```
public:virtual bool isControl() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are control characters.

Control characters are defined by the iswcntrl() C Library function as defined in the cntrl locale source file in the cntrl class of the LC_CTYPE category of the current locale. For example, on ASCII operating systems, control characters are those in the range {0x00-0x1F,0x7F}.

isDigits

```
public:virtual bool isDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9'}, true is returned.

isGraphics

```
public:virtual bool isGraphics() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are graphics characters.

Graphics characters are printable characters excluding the space character, as defined by the iswgraph() C Library function in the graph locale source file and in the graph class of the LC_CTYPE category of the current locale. For example, on ASCII operating systems, graphics characters are those in the range {0x21-0x7E}.

isHexDigits

```
public:virtual bool isHexDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9','A'-'F','a'-'f'}, true is returned.

isLowerCase

```
public:virtual bool isLowerCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'a'-'z'}, true is returned.

isPrintable

```
public:virtual bool isPrintable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are printable characters.

Printable characters are defined by the iswprint() C Library function as defined in the print locale source file and in the print class of the LC_CTYPE category of the current locale. For example, on ASCII systems, printable characters are those in the range {0x20-0x7E}.

isPunctuation

```
public:virtual bool isPunctuation() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If none of the characters is white space, a control character, or an alphanumeric character, true is returned.

isUpperCase

```
public:virtual bool isUpperCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z'}, true is returned.

isWhiteSpace

```
public:virtual bool isWhiteSpace() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are white-space characters.

White-space characters are defined by the iswspace() C Library function as defined in the space locale source file and in the space class of the LC_CTYPE category of the current locale. For example, on ASCII systems, printable white-space characters are those in the range {0x09-0x0D,0x20}.

IDBCSBuffer - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

IBuffer			
Definition	**Page Number**	**Definition**	**Page Number**
asDebugInfo	355	center	355
change	356	checkAddition	361
checkMultiplication	361	contents	360
copy	356	defaultBuffer	360
fromContents	360	insert	356
lastIndexOf	364	leftJustify	356
length	360	lowerCase	357
newBuffer	361	null	361
overflow	362	overlayWith	357
remove	357	reverse	357
rightJustify	357	setDefaultBuffer	362
strip	358	subString	365
translate	358	upperCase	358
useCount	361		

Inherited Public Data

None

Inherited Protected Functions

IBuffer			
Definition	**Page Number**	**Definition**	**Page Number**
~IBuffer	354	allocate	354
className	359	IBuffer	354
initialize	358	isDBCSLead	367
operator delete	354	operator new	355

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted	57	IMRefCounted	57
operator =	57		

Inherited Protected Data

IBuffer			
Definition	**Page Number**	**Definition**	**Page Number**
is32bytesAligned	368		

IDecimalFormat

IDecimalFormat provides functions to format and extract numbers in the decimal number system. Decimal numbers are base-10 numbers represented by symbols '0'...'9'. To format a number into a string, you can do the following:

```
IText dn = IDecimalFormat::format(10);
// dn = "10"
```

IDecimalFormat also supports locale-sensitive number formatting. By default, the current locale is used. You can override the default by supplying an ILocaleKey object or explicitly calling setlocale().

IDecimalFormat also supports parsing a decimal string into its numeric value. This is accomplished through a number of parse member functions. Each parse function returns the result in the requested type. IDecimalFormat does not attempt to guess the right type of number encoded in the argument.

See also:

- ILocaleKey
- IBinaryFormat
- IHexadecimalFormat

The following convenience functions convert a number string from one number system to another.

IText d2x(const IText& txt)
Convert a decimal string to a hexadecimal string.

IText x2d(const IText& txt)
Convert a hexadecimal string to a decimal string.

IText b2x(const IText& txt)
Convert a binary string to a hexadecimal string.

IText x2b(const IText& txt)
Convert a hexadecimal string to a binary string.

IText d2b(const IText& txt)
Convert a decimal string to a binary string.

IText b2d(const IText& txt)
Convert a binary string to a decimal string.

The following functions are only available if the compiler's long long support is turned on. By default, it is on.

IText d2xll(const IText& txt)
Convert a long long decimal string to a hexadecimal string.

IText x2dll(const IText& txt)
Convert a long long hexadecimal string to a decimal string.

IText b2xll(const IText& txt)
Convert a long long binary string to a hexadecimal string.

IText x2bll(const IText& txt)
Convert a long long hexadecimal string to a binary string.

IText d2bll(const IText& txt)
Convert a long long decimal string to a binary string.

IText b2dll(const IText& txt)
Convert a long long binary string to a decimal string.

Class header file: inumfrmt.hpp

IDecimalFormat - Hierarchy List

IDecimalFormat

IDecimalFormat - Member Functions and Data by Group

Formatting Decimal Number

Use these member functions to format a number into a decimal string.

format

Formats a number into a decimal string.

Overload 1

```
public:static IText
  format( long double n,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The long double value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

Overload 2

```
public:static IText
  format( double n,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The double value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

Overload 3

```
public:static IText
  format( long long n,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **AIX OS/400 z/OS**

n The long long value to format.

key The locale key used to perform locale-sensitive number formatting.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 4

```
public:static IText
  format( long number,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

number
The long value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

Overload 5

```
public:static IText
  format( long long n,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **AIX OS/400 z/OS**

n The long long value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 6

```
public:static IText
  format( unsigned long long n,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **AIX OS/400 z/OS**

n The unsigned long long value to format.

key The locale key used to perform locale-sensitive number formatting.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 7

```
public:static IText
  format( unsigned long long n,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **AIX OS/400 z/OS**

n The unsigned long long value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 8

```
public:static IText
  format( unsigned long n,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The unsigned long value to format.

key The locale key used to perform locale-sensitive number formatting.

Overload 9

```
public:static IText
  format( unsigned long n,
          const IText& pattern,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The unsigned long value to format.

pattern
A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

key The locale key used to perform locale-sensitive number formatting.

Overload 10

```
public:static IText
  format( long number,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

number
The long value to format.

key The locale key used to perform locale-sensitive number formatting.

Overload 11

```
public:static IText
  format( long double n,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The long double value to format.

key The locale key used to perform locale-sensitive number formatting.

Overload 12

```
public:static IText
  format( double n,
          const ILocaleKey& key = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The double value to format.

key The locale key used to perform locale-sensitive number formatting.

Parsing Decimal String

Use these member funcitons to parse a decimal string and return its numeric value.

parseDouble

```
public:static double parseDouble(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

parseLong

```
public:static long parseLong(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

parseLongDouble

```
public:static long double parseLongDouble(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

parseLongLong

```
public:static long long parseLongLong(const IText& txt)
```

This is supported on **AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

parseUnsignedLong

```
public:static unsigned long parseUnsignedLong(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

parseUnsignedLongLong

```
public:static unsigned long long
  parseUnsignedLongLong( const IText& txt )
```

This is supported on **AIX OS/400 z/OS**

Parses a decimal string and returns its numeric value. The argument should contain only valid decimal symbols. Otherwise, results are undefined.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

IDecimalFormat - Associated Globals

setLocale

```
static IText setLocale(const char* loc = 0)
```

Sets the locale.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDecimalFormat - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IFastTextIterator

The IFastTextIterator provides a high-performance forward iterator for IText. This class is similiar to ITextIterator, but is designed for higher performance. ITextIterator is designed to know about the styles and the reference-counting scheme and to do things in a way that works correctly with both. In addition, an ITextIterator has to be persistent. For example inserting text character by character with an iterator like the following example has to work.

```
while (thisIter != thisEnd)
 that.insert(thatIter++, *thisIter++);
```

These requirements place a performance burden on ITextIterator, so we added another class, IFastTextIterator, that is not bound by these requirements. It ignores reference-counting so you need to ensure that the IText being iterated across does not share storage with any other ITexts if you want to change the characters with an IFastTextIterator. You can do this by passing true for the constructor's willWrite parameter. This class also ignores styles, so you have to take that into account if you change the characters with an IFastTextIterator.

This class follows the protocol for an STL-compatible random-access iterator, and can be used with STL routines.

Special considerations:

- If you use an IFastTextIterator for writing, you must initialize it passing true for its willWrite parameter (this is the default). If you pass false for willWrite, but still write through the iterator, you may accidentally change other ITexts as well.
- IFastTextIterator does not do anything to styles. You must keep this in mind when using an IFastTextIterator to make changes to a styled IText.

- Do not initialize another IText from the IText under iteration or assign the IText under iteration to another IText while an IFastTextIterator is in effect. This could cause changes to the IText under iteration to affect the other IText.
- Do not call any non-const methods (or c_str() or data()) on the IText under iteration while an IFastTextIterator is in effect. The iterator is not guaranteed to remain valid across such calls.
- Do not derive from this class.

Class header file: itxtiter.hpp

IFastTextIterator - Hierarchy List

IFastTextIterator

IFastTextIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IFastTextIterator objects.

~IFastTextIterator

```
public:~IFastTextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the IFastTextIterator object.

IFastTextIterator

Constructs an IFastTextIterator object.

Overload 1

```
public:IFastTextIterator( const ITextIterator& that,
                          bool willWrite = true )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to creates an IFastTextIterator that refers to the same character as the iterator passed in. This enables expressions like the following:

```
IFastTextIterator iter = (IFastTextIterator)(text.begin());
```

that The ITextIterator whose iterator position is copied.

willWrite
If true, constructing the iterator forces the IText under iteration traversal to guarantee that its storage is not shared, preventing changes made through the iterator from accidentally affecting other IText objects. If false, nothing special is done to the IText. This indicates a promise on the caller's behalf to use the iterator for read-only access. The default is true.

Overload 2

```
public:IFastTextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this default constructor to create an IFastTextIterator object that leaves the iterator in an undefined state. The assignment operator must be called before a default-constructed iterator can be used.

Overload 3

```
public:IFastTextIterator( const IConstTextIterator& that,
                  bool willWrite = true )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IFastTextIterator object that refers to the same character as the iterator passed in. This enables expressions such as the following:

```
IFastTextIterator iter = (IFastTextIterator)(text.begin());
```

that The IConstTextIterator whose iterator position is copied.

willWrite
If true, constructing the iterator forces the IText under iteration traversal to guarantee that its storage is not shared, preventing changes made through the iterator from accidentally affecting other IText objects. If false, nothing special is done to the IText. This indicates a promise on the caller's behalf to use the iterator for read-only access. The default is true.

Overload 4

```
public:IFastTextIterator(const IFastTextIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an object of class IFastTextIterator which is copied from the given one.

that The IFastTextIterator that is copied.

Assignment Operator

Use this operator to replace the current IFastTextIterator with the given one.

operator =

```
public:IFastTextIterator&
  operator =( const IFastTextIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Points this, the current, iterator to the same character position in the same IText as the one given by that points to.

that The IFastTextIterator object whose value replaces the current one.

Comparisons

Use these operators to obtain the number of character positions between the two iterators or to compare the positions of the given iterator and the current one.

operator !=

```
public:bool operator !=(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the iterators refer to different character positions in the same IText. This is a straight pointer comparison. No error checking is performed.

operator -

```
public:ioc::numeric_t
  operator -( const IFastTextIterator& that ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of character positions between the two iterators. This is a simple pointer subtraction; there is no error checking.

that The IFastTextIterator object whose iterator position is compared with the position of the current one.

operator <

```
public:bool operator <(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this, the current iterator, refers to a character with a lower offset than does the given one. This operator performs a straight pointer comparison, there is no error checking.

that The IFastTextIterator object whose iterator position is compared with the position of the current one.

operator <=

```
public:bool operator <=(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this, the current iterator, refers to the same character as does the given one or if it refers to one with a lower offset. This operator performs a straight pointer comparison, there is no error checking.

that The IFastTextIterator object whose iterator position is compared with the position of the current one.

operator ==

```
public:bool operator ==(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if both iterators refer to the same character position in the same IText. This is a straight pointer comparison. No error checking is performed.

operator >

```
public:bool operator >(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this, the current iterator, refers to a character with a higher offset than does the current one. This operator performs a straight pointer comparison, there is no error checking.

that The IFastTextIterator object whose iterator position is compared with the position of the current one.

operator >=

```
public:bool operator >=(const IFastTextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this, the current iterator, refers to the same character as does the given one or if it refers to one with a higher offset. This operator performs a straight pointer comparison, there is no error checking.

that The IFastTextIterator object whose iterator position is compared with the position of the current one.

Incrementing and Decrementing

Use these operators to increment or decrement the iterator's position.

operator ++

Advances the iterator to the next character position, that is, the position with the next-higher offset.

Overload 1

```
public:IFastTextIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the iterator before it was advanced.

Overload 2

```
public:IFastTextIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to this.

operator +=

```
public:IFastTextIterator& operator +=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator the number of character positions toward the end of the text, that is, toward higher character offsets, specified by distance. Returns a reference to this.

operator -

```
public:IFastTextIterator operator -(ioc::numeric_t distance) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator --

Advances the iterator to the previous character position, that is, the position with the next lower offset.

Overload 1

```
public:IFastTextIterator operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the iterator before it was advanced.

Overload 2

```
public:IFastTextIterator& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to this.

operator -=

```
public:IFastTextIterator& operator -=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator toward the beginning of the text, that is, toward lower character offsets, the number of character positions specified by distance. Returns a reference to this.

Obtaining a Reference to a Character

Use these member functions to obtain a reference to a character whose position the iterator refers to.

operator *

```
public:ioc::unichar_t& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the character the iterator refers to. This is a straight pointer dereference; there is no error checking.

operator []

```
public:ioc::unichar_t& operator [](ioc::numeric_t offset) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the character the number of positions, specified by offset, away from the one the iterator currently refers to. Here is an example of how this iterator operator works: if iter points to position 2 in your function, iter[3] will return the character at position 5, not the character at position 3. The iterator itself is left unchanged by this operation. This is a straight pointer dereference; there is no error checking.

offset The displacement or number of positions away from the position of the character the iterator currently refers to.

IFastTextIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IHexadecimalFormat

IHexadecimalFormat provides functions to format and extract numbers in the hexadecimal number system. Hexadecimal numbers are base-16 numbers. Letters A, B, C, D, E, and F are used to represent 10, 11, 12, 13, 14, and 15 respectively. To format a number in the hexadecimal number system, you use the format member function:

```
IText bn = IHexadecimalFormat::format(10);
// bn = "A"
```

To convert a hexadecimal string back into its numeric value. you can use the parseUnsignedLong() or the parseUnsignedLongLong() function. The latter is only supported if the compiler's long long support is turned on. By default, it is on. For example:

```
unsigned long i = IHexadecimalFormat::parseUnsignedLong("A");
// i = 10
```

See also:

- IBinaryFormat
- IDecimalFormat

The following convenience functions convert a number string from one number system to another.

IText d2x(const IText& txt)
Convert a decimal string to a hexadecimal string.

IText x2d(const IText& txt)
Convert a hexadecimal string to a decimal string.

IText b2x(const IText& txt)
Convert a binary string to a hexadecimal string.

IText x2b(const IText& txt)
Convert a hexadecimal string to a binary string.

IText d2b(const IText& txt)
Convert a decimal string to a binary string.

IText b2d(const IText& txt)
Convert a binary string to a decimal string.

The following functions are only available if the compiler's long long support is turned on. By default, it is on.

IText d2xll(const IText& txt)
Convert a long long decimal string to a hexadecimal string.

IText x2dll(const IText& txt)
Convert a long long hexadecimal string to a decimal string.

IText b2xll(const IText& txt)
Convert a long long binary string to a hexadecimal string.

IText x2bll(const IText& txt)
Convert a long long hexadecimal string to a binary string.

IText d2bll(const IText& txt)
Convert a long long decimal string to a binary string.

IText b2dll(const IText& txt)
Convert a long long binary string to a decimal string.

Class header file: inumfrmt.hpp

IHexadecimalFormat - Hierarchy List

IHexadecimalFormat

IHexadecimalFormat - Member Functions and Data by Group

Formatting Hexadecimal String

Use these member functions to format a number into a hexadecimal string.

format

Formats a number into a hexadecimal string.

Overload 1

```
public:static IText format(unsigned long n, const IText& pattern)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The unsigned long value to format.

pattern

A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

AIX Considerations

Unicode format strings are not supported.

z/OS Considerations

Unicode format strings are not supported.

Overload 2

```
public:static IText
  format( unsigned long long n,
          const IText& pattern )
```

This is supported on **AIX OS/400 z/OS**

n The unsigned long long value to format.

pattern

A format string with which to format the number. This format string uses the same syntax as the C's sprintf format string.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

AIX Considerations

Unicode format strings are not supported.

z/OS Considerations

Unicode format strings are not supported.

Overload 3

```
public:static IText format(unsigned long n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

n The unsigned long value to format.

Overload 4

```
public:static IText format(unsigned long long n)
```

This is supported on **AIX OS/400 z/OS**

n The unsigned long long value to format.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Parsing Hexadecimal Number

Use these member functions to parse a hexadecimal string and return its numeric value.

parseUnsignedLong

```
public:static unsigned long parseUnsignedLong(const IText& txt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses a hexadecimal string and returns its numeric value. The argument should contain only valid hexadecimal symbols. Otherwise, results are undefined.

parseUnsignedLongLong

```
public:static unsigned long long
  parseUnsignedLongLong( const IText& txt )
```

This is supported on **AIX OS/400 z/OS**

Parses a hexadecimal string and returns its numeric value. The argument should contain only valid hexadecimal symbols. Otherwise, results are undefined.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

IHexadecimalFormat - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ILineBreakConverter

The ILineBreakConverter class is used to process the line breaking conventions for code set conversion.

ILineBreakConverter processes the line breaking conventions for code set conversion. When a foreign code set string is transcoded to Unicode, the foreign code set line breaks are not converted. That means a transcoded Unicode string still carries the foreign code set line breaks. Clients are required to postprocess this Unicode string to using the Unicode line breaking convention (for example, the PARAGRAPH SEPARATORs -- U+2029). ILineBreakConverter is a simple concrete class that does this. For instance, on DOS, Win32, and OS/2 hosts, the line breaks (CR/LF sequences) will be converted to PARAGRAPH SEPARATORs. On the AIX platform, LFs will be converted to PARAGRAPH SEPARATORs.

Converting the PARAGRAPH SEPARATOR back to an appropriate line breaking convention depends on the host on which the document is being viewed or used. The following table shows the rules used during the line breaks conversion:

```
Host/App            Host Line Breaks                  Unicode Line Breaks
========            ================                  ===================
DOS,OS/2,Win       CR/LF:0x000D 0x000A           <=> U+2029 PARAGRAPH SEPARATOR
Unix                    LF:0x000A                <=> U+2029 PARAGRAPH SEPARATOR
Mac                     CR:0x000D                <=> U+2029 PARAGRAPH SEPARATOR
Word/RichEdit      CR/LF_VT:0x000D 0x000A        <=> U+2029 PARAGRAPH SEPARATOR
Word/RichEdit      CR/LF_VT:0x000B               <=> U+2028 LINE SEPARATOR
```

To use this class, the following scenario shows how to postprocess the line breaks after a text string is transcoded to Unicode, and then preprocess the line breaks before transcoding the text string back to host character set:

```
transcoder->toUnicode(hostText, uniText);
ILineBreakConverter::convertInPlace(uniText); // postprocess
// Do something on uniText ...
ILineBreakConverter::convertInPlace(uniText, ILineBreakConverter::kHost); // preprocess
transcoder->fromUnicode(uniText, hostText);
```

You should not derive from this class.

Class header file: itrancod.hpp

ILineBreakConverter - Hierarchy List

ILineBreakConverter

ILineBreakConverter - Member Functions and Data by Group

Conversion Members

Use these member functions to convert a Unicode string using the specified line break convention. Use these functions to postprocess and preprocess the line breaks after and before the text is converted to the foreign host character set.

convert

```
public:static IText
  convert( const IText& string,
           ELineBreakConvention target = kUnicode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts a Unicode string to using the target line break convention. The default uses the Unicode line break convention. Returns a Unicode string containing the target line break convention.

Note: Use this function to postprocess the line breaks for the string after it is converted to Unicode, and preprocess the line breaks before it is converted back to foreign host character set.

string The given Unicode string.

target The target line break convention.

convertInPlace

```
public:static void
  convertInPlace( IText& string,
                  ELineBreakConvention target = kUnicode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts a Unicode string in place to using target line break convention. The default uses the Unicode line break convention.

Note: Use this function to postprocess the line breaks after converting to Unicode, and preprocess the line breaks before converting back to foreign host character set.

string The given Unicode string.

target The target line break convention.

Host Conventions

Use this member function to obtain the current line breaking convention used for the host.

hostConvention

```
public:static ELineBreakConvention hostConvention()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current host line break convention.

ILineBreakConverter - Enumerations

ELineBreakConvention

Useful constants indicating the target line break convention.

```
kUnicode - Unicode line break convention.
kCRLF    - Win, OS/2, DOS line break convention.
kLF      - UNIX line break convention.
kCR      - Macintosh line break convention.
kCRLF_VT - Microsoft Word/RichEdit line break convention.
kHost    - Current host line break convention.
```

ILineBreakConverter - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ILocaleKey

ILocaleKey class provides an access mechanism for any localizable resources.

ILocaleKey class provides an access mechanism for any localizable resources. The purpose of the locale key component is to provide support for identifying linguistic and local conventions.

Since locales are only a mechanism that allows programmers to access certain resources associated with a given country or region, the ILocaleKey itself is not a set of objects. Rather, it is a key that can be used to access a particular international object. Many international objects can be created using a locale. As a convenience, there is a default locale that is used when none is specified. The locale key mechanism is intended to be used with Unicode text manipulation.

Note: International objects are those whose data/code may depend on the country or language, and which are used to support globalized programs. Globalized programs are those written so that they work no matter what language or country they are used with. Examples of international objects are collation, date, time, or number classes.

Do not derive your own classes from this class.

Class header file: ilocale.hpp

ILocaleKey - Hierarchy List

IMStreamable

ILocaleKey

ILocaleKey - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ILocaleKey objects.

~ILocaleKey

```
public:~ILocaleKey()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ILocaleKey object.

ILocaleKey

Constructs an ILocaleKey object.

Overload 1

```
public:ILocaleKey(ioc::unumeric_t hostLocaleID)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ILocaleKey( const IText& languageID,
            const IText& regionID = IText ( ),
            const IText& ext = IText ( ),
            const IText& path = IText ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ILocaleKey(const ILocaleKey& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes an ILocaleKey object from another ILocaleKey object.
that The ILocaleKey object being copied in.

Overload 4

```
public:ILocaleKey()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a default locale key.

Accessing ILocaleKey Information

Use these member functions to get and set the entire POSIX ID, which is composed of a language ID, region ID and extension, or any of its parts. Using the functions in this group, you can also get and set the host locale ID of the ILocaleKey object and determine whether the current locale is the POSIX C or POSIX locale.

extension

```
public:IText extension() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the POSIX extension of the locale key.

hostID

```
public:ioc::unumeric_t hostID() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the host locale ID of this ILocaleKey object. The returned ID is the Window's LCID.

isCLocale

```
public:bool isCLocale() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if the current locale is the POSIX C or POSIX locale.

languageID

```
public:IText languageID() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the POSIX language ID of the locale key.

path

```
public:IText path() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

posixID

```
public:IText posixID() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the POSIX ID, which is composed of a language ID, region ID and extension.

regionID

```
public:IText regionID() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the POSIX region ID of the locale key.

setExtension

```
public:void setExtension(const IText& extension)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the current locale POSIX extension to the new extension.

IText A constant reference to the new extension.

Exception

Exception	Description
IInvalidParameter	If the new POSIX ID is not a valid locale ID, set the POSIX ID to be the valid locale with the same language ID (the region ID and extension will be ignored). If no corresponding locale can be found, throws the IInvalidParameter exception.
IInvalidRequest	If the new POSIX ID of the matched locale is supported but not installed on the machine; there is no associated international resources, an IInvalidRequest exception will be thrown.

setHostID

```
public:void setHostID(ioc::unumeric_t localeID)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Set the host locale ID.

localeID

The new host locale ID.

setLanguageID

```
public:void setLanguageID(const IText& languageID)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the current locale POSIX language ID to the new language ID.

languageID

A constant reference to the new language ID.

Exception

Exception	Description
IInvalidParameter	If the new POSIX ID is not a valid locale ID, set the POSIX ID to be the valid locale with the same language ID (the region ID and extension will be ignored). If no corresponding locale can be found, throws the IInvalidParameter exception.
IInvalidRequest	If the new POSIX ID of the matched locale is supported but not installed on the machine; there is no associated international resources, an IInvalidRequest exception will be thrown.

setPath

```
public:void setPath(const IText& path)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

setPOSIXID

```
public:void setPOSIXID(const IText& localeID)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the POSIX ID, which is composed of a language ID, region ID and extension.

localeID

A constant reference to the new locale ID.

Exception

Exception	Description
IInvalidParameter	If the POSIX ID is not a valid locale ID, set the POSIX ID to be the valid locale with the same language ID (the region ID and extension will be ignored). If no corresponding locale can be found, throws the IInvalidParameter exception.
IInvalidRequest	If the POSIX ID of the matched locale is supported but not installed on the machine; there is no associated international resources, an IInvalidRequest exception will be thrown.

setRegionID

```
public:void setRegionID(const IText& regionID)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the current locale POSIX region ID to the new region ID.

regionID

A constant reference to the new region ID.

Exception

Exception	Description
IInvalidParameter	If the new POSIX ID is not a valid locale ID, set the POSIX ID to be the valid locale with the same language ID (the region ID and extension will be ignored). If no corresponding locale can be found, throws the IInvalidParameter exception.
IInvalidRequest	If the new POSIX ID of the matched locale is supported but not installed on the machine; there is no associated international resources, an IInvalidRequest exception will be thrown.

Assignment Operator

Use this operator to replace the current locale key object with the given one.

operator =

```
public:ILocaleKey& operator =(const ILocaleKey& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents of *this with the specified value.

that The ILocaleKey object use to replace the current one.

Comparing ILocaleKey Objects

Use these operators to test the given ILocaleKey object against the current one for equality or inequality.

operator !=

Checks if two locale keys are not the same.

```
public:bool operator !=(const ILocaleKey& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given locale key object against the current one to determine if they are different.

other The locale key object to be compared with this.

Return

true if the two locale keys are not the same, false otherwise.

operator ==

Checks if two locale keys are the same.

```
public:bool operator ==(const ILocaleKey& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given locale key object against the current one to determine if they are the same.

other The locale key object to be compared with this.

Return

true if the two locale keys are the same, false otherwise.

Creating an ILocaleKey Object from the Default Locale

Use this member function to create a locale key object based on the default locale.

defaultLocale

```
public:static ILocaleKey defaultLocale()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a locale key object based on the default locale.

Return

the default user-defined locale for the thread if there is one. Returns the thread's system default locale otherwise.

Displayable Name

Use these member functions to obtain the locale's displayable based on the given key, language, and search values.

displayName

Returns the displayable name of a selected locale based on the lookup strategy. If there is no display name for the locale and the locale has associated international resources, this class throws the IOutOfSystemResource exception. If the locale is supported but not installed, this class throws the IInvalidRequest exception. If the locale ID is invalid, this class throws the IInvalidParameter exception.

Overload 1

```
public:IText
  displayName( const ILocaleKey& desiredLanguage = defaultLocale ( ),
               ELookupStrategy strategy = kAnyLanguage ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this member function to obtain the locale's displayable name given the desired language used to express the name, and a lookup strategy.

desiredLanguage

A constant reference to the language of the locale, which is used express its displayable name.

strategy

The strategy used to look up the display name. The default value is kAnyLanguage.

Return

the displayable name of the current locale.

Exception

Exception	Description
IOutofSystemResource	If there is no display name for the given locale and the locale has associated international resources.
IInvalidRequest	If the locale is supported but not installed.
IInvalidParameter	If the locale ID is invalid.

Overload 2

```
public:static IText
  displayName( const IText& key,
               const ILocaleKey& desiredLanguage = defaultLocale ( ),
               ELookupStrategy strategy = kAnyLanguage )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this member function to obtain the locale's displayable name, given a key, the desired language used to express the name, and a lookup strategy.

key The POSIX ID to get the displayable name with

desiredLanguage

A constant reference to the locale used to display the name. in.

strategy

The look-up strategy for the display name of the POSIX ID. The default value is kAnyLanguage.

Return

the displayable name of the POSIX ID.

Exception

Exception	Description
IOutofSystemResource	If there is no display name for the given locale and the locale has associated international resources.
IInvalidRequest	If the locale is supported but not installed.
IInvalidParameter	If the locale ID is invalid.

Streaming ILocaleKey Objects

Use these member functions to read the ILocaleKey object in from the given stream or write it out to it.

readFromStream

```
protected:virtual void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

fromWhere

Where the locale ID will be read from.

writeToStream

```
protected:virtual void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

toWhere
Where the locale ID will be written to.

ILocaleKey - Enumerations

ELookupStrategy

ELookupStrategy is used to specify the look-up strategy for a locale key display name.

Use these constants enumerated by ELookupStrategy to specify the lookup strategy for a locale key display name:

kExactLocale
The display name of the locale key for the exact given locale ID will be looked up.

kExactLanguage
The display name of the locale key for the given locale ID will be returned. If there is no display name for the exact given locale ID, the display name for the given language will be looked up. It will consider only the language portion of the language and the region ID will be ignored.

kAnyLanguage
If the display name cannot be found for the exact given locale ID nor the given language, displayName() will look for the display name in the default locale. This is the default display name look-up strategy.

ILocaleKey - Inherited Member Functions and Data

Inherited Public Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
~IMStreamable	200	externalizeToStream	201
internalizeFromStream	201	typeRepresentation	201

Inherited Public Data

None

Inherited Protected Functions

IMStreamable			
Definition	**Page Number**	**Definition**	**Page Number**
IMStreamable	200	readFromStream	201
writeToStream	202		

Inherited Protected Data

None

ILocaleKeyIterator

The ILocaleKeyIterator class lets you iterate through the locales that are currently installed on the local machine which have resources associated with them.

ILocaleKeyIterator provides the protocol for traversing the list of all available locales that have international resources associated with them. The list traversed by the iterator is the one on the current machine. As such, it may be a subset of all supported locales. For this reason, the returned data will not provide information about supported locales that are not installed.

You must specify the environment variable LOCPATH before the using POSIX international resources.

Do not derive your own classes from this class.

Class header file: ilocale.hpp

ILocaleKeyIterator - Hierarchy List

ILocaleKeyIterator

ILocaleKeyIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ILocaleKeyIterator objects.

~ILocaleKeyIterator

```
public:~ILocaleKeyIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ILocaleKeyIterator object.

ILocaleKeyIterator

Constructs an ILocaleKeyIterator object.

Overload 1

```
public:ILocaleKeyIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Exception

Exception	Description
IInvalidRequest	Thrown when instantiating the ILocaleKeyIterator using the POSIX interface and the LOCPATH variable is not specified.

Overload 2

```
public:ILocaleKeyIterator(const ILocaleKeyIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Initializes an ILocaleKeyIterator object from another ILocaleKeyIterator object.

that The ILocaleKeyIterator object being copied in.

Assignment Operator

Use this operator to replace the current ILocaleKeyIterator with the given one.

operator =

```
public:ILocaleKeyIterator&
  operator =( const ILocaleKeyIterator& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the entire contents of *this with the specified value.

that The ILocaleKeyIterator object being copied in.

Comparing ILocaleKeyIterator Objects

Use these operators to test the given locale key iterator against the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ILocaleKeyIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given iterator with the current one for inequality.

that A constant reference to the locale key iterator object to be compared with this, the current, one.

Return

true if the two locale key iterators are not the same, false otherwise.

operator ==

```
public:bool operator ==(const ILocaleKeyIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given iterator with the current one for equality.

that A constant reference to the locale key iterator object to be compared with this, the current, one.

Return

true if the two locale key iterators are the same, false otherwise.

Detecting the End of the Iterator

Use these member functions to determine if the iterator has traversed all the items of the list and has consumed the last one.

end

```
public:static ILocaleKeyIterator end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if the iterator has consumed the last list item.

Return

the equivalent of NIL atom of a list to show the last list item has been consumed.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if the iterator has consumed the last list item.
Return
true if there are no more locales to iterate through, false otherwise.

Getting the Current Locale Key

Use this member function to obtain the current locale key.
operator *

```
public:ILocaleKey operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current locale key in the list.

Incrementing the Iterator

Use this operator to increment the iterator to the next position.
operator ++

```
public:ILocaleKeyIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Operator that increments the locale key iterator.

Advances to the next locale key.
int This (int type) represents the postfix form of ++ operator.

Returns the advanced locale key iterator.
Exception

Exception	Description
IInvalidRequest	If the last locale key has been consumed and an advancing request is made by the clients.

Moving and Resetting the Cursor

Use these member functions to move and reset the cursor.
operator ++

```
public:ILocaleKeyIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Increments the iterator after the operation is performed, using the postfix increment operator. Returns the advanced locale key iterator.
Exception

Exception	Description
IInvalidRequest	If the last locale key has been consumed and an advancing request is made by the clients.

operator ->

```
public:const ILocaleKey* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a constant pointer to the current locale key in the list.
reset

```
public:void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reset the cursor pointer to the first item on the list.

ILocaleKeyIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMTabRuler

IMTabRuler is an abstract class that defines a tab-ruler protocol. IStandardTabRuler provides a concrete implementation for IMTabRuler.

Do not derive from this class.

Class header file: imtabrl.hpp

IMTabRuler - Hierarchy List

IMTabRuler

IStandardTabRuler

IMTabRuler - Member Functions and Data by Group

Assignment Operator

Use this operator to replace the current tab ruler with the given one.

operator ==

```
public:bool operator ==(const IMTabRuler& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Cloning the Tab Ruler

Use this member function to copy the tab ruler.

clone

```
public:virtual IMTabRuler* clone() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a new tab ruler that is identical to this one.

Obtaining a Copy of a Tab

Use these member functions to copy the tab at the specified position.

firstTab

```
public:virtual ITabStop firstTab() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the first tab in the ruler. If an autoTab, it is at position zero, and all subsequent tabs will be autotabs at autoSpacing intervals.

nextTab

```
public:virtual ITabStop nextTab(GCoord position) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the first tab in the ruler with fPosition > position. If it is an autotab, it is at an even multiple of autoSpacing, and all subsequent tabs will be autotabs at autoSpacing intervals.

Obtaining Information about Tab Stops

Use these member functions to obtain the number of tabs stops in the ruler or the interval between tab stops.

autoSpacing

```
public:virtual GCoord autoSpacing() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the interval for autotabs.

tabCount

```
public:virtual ioc::isize_t tabCount() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of tab stops in this ruler.

Obtaining References to Tab Stops

Use these member functions to obtain a constant or mutable reference to a specific tab stop.

operator []

Returns a modifiable reference to the ″ndx″-th tab stop in the ruler.

Overload 1

```
public:virtual ITabStop& operator [](ioc::isize_t ndx) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual const ITabStop&
  operator [](ioc::isize_t ndx ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMTabRuler - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IParagraphEndIndentStyle

The IParagraphEndIndentStyle class is a style that controls the ending indent distance of the paragraphs it is applied to. The value is a number of points, expressed as a float.

Do not derive your own classes from this class.

Class header file: istyles.hpp

IParagraphEndIndentStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphEndIndentStyle

IParagraphEndIndentStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphEndIndentStyle objects.

~IParagraphEndIndentStyle

```
public:virtual ~IParagraphEndIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphEndIndentStyle object.

IParagraphEndIndentStyle

Constructs an IParagraphEndIndentStyle object.

Overload 1

```
public:IParagraphEndIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 2

```
public:IParagraphEndIndentStyle(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create ab IParagraphEndIndentStyle object that is initialized with the current value.
value The amount of indentation.

Overload 3

```
public:IParagraphEndIndentStyle( const IParagraphEndIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphEndIndentStyle object that is copied from the given one.

other A constant reference to the IParagraphEndIndentStyle object to be copied.

Assignment Operator

Use this operator to replace the current paragraph end indent style object with the given one.

operator =

```
public:IParagraphEndIndentStyle&
  operator =( const IParagraphEndIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphEndIndentStyle object to assign.

Comparing IParagraphEndIndentStyle Objects

Use these operators to compare the given IParagraphEndIndentStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const IParagraphEndIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphEndIndentStyle object against the current one for inequality.

other A constant reference to the IParagraphEndIndentStyle to be compared with the current one.

operator ==

```
public:bool
  operator ==( const IParagraphEndIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphEndIndentStyle object against the current one for equality.

other A constant reference to the IParagraphEndIndentStyle to be compared with the current one.

Getting and Setting the End Indentation

Use these member functions to get and set the amount of indentation at the end of the paragraphs.

endIndent

```
public:float endIndent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current setting that determines the amount of indentation at end of the paragraphs.

setEndIndent

```
public:void setEndIndent(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the amount of indentation at end of the paragraphs for the current IParagraphEndIndentStyle.

value A floating point value that defines the amount of indentation.

IParagraphEndIndentStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

IParagraphEndIndentStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	453	getValueFromSet	453
isContainedInSet	453	isValueContainedInSet	453
removeFromSet	453		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

IParagraphFirstIndentStyle

The IParagraphFirstIndentStyle class is a style that controls the indent distance at the beginning of the paragraphs it is applied to. The value is a number of points, expressed as a float.

Do not derive your own classes from this class.

Class header file: istyles.hpp

IParagraphFirstIndentStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphFirstIndentStyle

IParagraphFirstIndentStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphFirstIndentStyle objects.

~IParagraphFirstIndentStyle

```
public:virtual ~IParagraphFirstIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphFirstIndentStyle object.

IParagraphFirstIndentStyle

Constructs an IParagraphFirstIndentStyle object.

Overload 1

```
public:IParagraphFirstIndentStyle( const IParagraphFirstIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the copy constructor to create an IParagraphFirstIndentStyle object that is copied from the given one.

other The IParagraphFirstIndentStyle object to be copied.

Overload 2

```
public:IParagraphFirstIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphFirstIndentStyle object that leaves the style in an undefined state. The value for the default-constructed style must be filled in before the obejct is used.

Overload 3

```
public:IParagraphFirstIndentStyle(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphFirstIndentStyle object that is initialized with the given indentation value.

value The amount of indentation, expressed in points, to be used for the beginning of the paragraphs.

Assignment Operator

Use this operator to replace the current IParagraphFirstIndentStyle object with the current one.

operator =

```
public:IParagraphFirstIndentStyle&
  operator =( const IParagraphFirstIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphFirstIndentStyle object to be used to replace this, the current, one.

Comparing IParagraphFirstIndentStyle Objects

Use these operators to compare the given IParagraphFirstIndentStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const IParagraphFirstIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphFirstIndentStyle object against the current one for inequality.

other The given IParagraphFirstIndentStyle object to be compared with the current one.

operator ==

```
public:bool
  operator ==( const IParagraphFirstIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphFirstIndentStyle object against the current one for equality.

other The IParagraphFirstIndentStyle object to be tested against this, the current, one.

Getting and Setting the Indentation Amount

Use these member functions to get and set the amount of indentation to be used at the beginning of a paragraph.

firstIndent

```
public:float firstIndent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current setting for indentation at the beginning of a paragraph.

setFirstIndent

```
public:void setFirstIndent(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the amount of indentation to be applied to the beginning of a paragraph.

value The amount of space in points, expressed as a float.

IParagraphFirstIndentStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

IParagraphFirstIndentStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	456	getValueFromSet	456
isContainedInSet	456	isValueContainedInSet	456
removeFromSet	456		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

IParagraphJustificationStyle

The IParagraphJustificationStyle class is a style that controls the justification direction of the paragraphs it is applied to.

Do not derive your own classes from this class.

Class header file: istyles.hpp

IParagraphJustificationStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphJustificationStyle

IParagraphJustificationStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphJustificationStyle objects.

~IParagraphJustificationStyle

```
public:virtual ~IParagraphJustificationStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphJustificationStyle object.

IParagraphJustificationStyle

Constructs an IParagraphJustificationStyle object.

Overload 1

```
public:IParagraphJustificationStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 2

```
public:IParagraphJustificationStyle(EJustification value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphJustificationStyle object that is initialized to the value specified by the given EJustification constant.

value An EJustification enumerated constant specifying the type of justification.

Overload 3

```
public:IParagraphJustificationStyle( const IParagraphJustificationStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IParagraphJustificationStyle object that is copied from the given one.

other The IParagraphJustificationStyle object to be copied.

Assignment Operator

Use this assignment operator to replace the current IParagraphJustificationStyle object with the given one.

operator =

```
public:IParagraphJustificationStyle&
  operator =( const IParagraphJustificationStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphJustificationStyle object to be used to replace this, the current, one.

Comparing IParagraphJustificationStyle Objects

Use these member functions to compare the given IParagraphJustificationStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const IParagraphJustificationStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphJustificationStyle object against the current one for inequality.

other The IParagraphJustificationStyle to be tested against this, the current, one.

operator ==

```
public:bool
  operator ==( const IParagraphJustificationStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphJustificationStyle object with the current one for equality.

other The IParagraphJustificationStyle to be tested against this, the current, one.

Getting and Setting the Justification Value

Use these member functions to get and set the justification value.

justification

```
public:EJustification justification() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the justification value.

setJustification

```
public:void setJustification(EJustification value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the justification value using the given enumerated constant.

value An EJustification enumerated constant specifying the type of justification/

IParagraphJustificationStyle - Enumerations

EJustification

Constant values specifying different justification directions

Use these enumerated constants to specify different justification directions:

kStart Aligns each line to the ″beginning″ of the line in the current writing direction, for example, for Roman text, align to the left margin, but for Hebrew text, align to the right margin.

kEnd Align each line to the end of the line in the current writing direction, for example, for Roman text, align to the right margin, but for Hebrew text, align to the left margin of the paragraph between the margins

IParagraphJustificationStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

IParagraphJustificationStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	459	getValueFromSet	459
isContainedInSet	459	isValueContainedInSet	459
removeFromSet	459		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ITextStyle removeFromSet	632 633	operator =	633

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

IParagraphSpreadStyle

The IParagraphSpreadStyle class is a style that controls which lines of the paragraphs it is applied to (if any) are fully justified (aligned to both the left and right margins).

Do not derive your own classes from this class.

Class header file: istyles.hpp

IParagraphSpreadStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphSpreadStyle

IParagraphSpreadStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphSpreadStyle objects.

~IParagraphSpreadStyle

```
public:virtual ~IParagraphSpreadStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphSpreadStyle object.

IParagraphSpreadStyle

Constructs an IParagraphSpreadStyle object.

Overload 1

```
public:IParagraphSpreadStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphFirstIndentStyle object that leaves the style in an undefined state. The value for the default-constructed style must be filled in before the obejct is used.

Overload 2

```
public:IParagraphSpreadStyle(const IParagraphSpreadStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the copy constructor to create an IParagraphSpreadStyle object that is copied from the given one.

other The IParagraphSpreadStyle object to be copied.

Overload 3

```
public:IParagraphSpreadStyle(ESpread value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphSpreadStyle object that is initialized with the given spread value.

value The amount of spread (which lines of the paragraphs are aligned to both the left and right margins).

Assignment Operator

Use this operator to replace the current IParagraphSpreadStyle object with the current one.

operator =

```
public:IParagraphSpreadStyle&
  operator =( const IParagraphSpreadStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphSpreadStyle object to be used to replace this, the current, one.

Comparing IParagraphSpreadStyle Objects

Use these operators to compare the given IParagraphSpreadStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const IParagraphSpreadStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphSpreadStyle object against the current one for inequality.

other The given IParagraphSpreadStyle object to be compared with the current one.

operator ==

```
public:bool operator ==(const IParagraphSpreadStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphSpreadStyle object against the current one for equality.

other The IParagraphSpreadStyle object to be tested against this, the current, one.

Getting and Setting the Spread Value

Use these member functions to get and set the amount of spread.

setSpread

```
public:void setSpread(ESpread value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the spread value (which lines of the paragraphs are aligned to both the left and right margins).

spread

```
public:ESpread spread() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the spread value (which lines of the paragraphs are aligned to both the left and right margins).

IParagraphSpreadStyle - Enumerations

ESpread

Constant definitions specifying the lines of the paragraph to fully justify:

kNone
All lines are aligned according to the IParagraphJustificationStyle, and the opposite end of the line is left ragged.

kAll All lines are aligned to both margins (i.e., white space is added in the middle as necessary), and the IParagraphJustificationStyle is ignored.

kNotLastLine
All lines of the paragraph except for the last one are aligned to both the left and right margins; the alignment of the last line is controlled by the IParagraphJustificationStyle.

IParagraphSpreadStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

IParagraphSpreadStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	462	getValueFromSet	462
isContainedInSet	462	isValueContainedInSet	462
removeFromSet	462		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

IParagraphStartIndentStyle

The IParagraphStartIndentStyle class is a style that controls the starting indent distance of the paragraphs it is applied to. The value is a number of points, expressed as a float.

Do not derive your own classes from this class.

Class header file: istyles.hpp

IParagraphStartIndentStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphStartIndentStyle

IParagraphStartIndentStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphStartIndentStyle objects.

~IParagraphStartIndentStyle

```
public:virtual ~IParagraphStartIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphStartIndentStyle object.

IParagraphStartIndentStyle

Constructs an IParagraphStartIndentStyle object.

Overload 1

```
public:IParagraphStartIndentStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 2

```
public:IParagraphStartIndentStyle( const IParagraphStartIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IParagraphStartIndentStyle object that is copied from the given one.

other The IParagraphStartIndentStyle object to be copied.

Overload 3

```
public:IParagraphStartIndentStyle(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphStartIndentStyle object that is initialized to the starting indentation value.

value The starting indentation value. The value is a number of points, expressed as a float.

Assignment Operator

Use this assignment operator to replace the current IParagraphStartIndentStyle object with the given one.

operator =

```
public:IParagraphStartIndentStyle&
  operator =( const IParagraphStartIndentStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphStartIndentStyle object to be used to replace this, the current, one.

Comparing IParagraphStartIndentStyle Objects

Use these member functions to compare the given IParagraphStartIndentStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const IParagraphStartIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphStartIndentStyle object with the current one for inequality.

other The IParagraphStartIndentStyle to be tested against this, the current, one.

operator ==

```
public:bool
  operator ==( const IParagraphStartIndentStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given IParagraphStartIndentStyle object with the current one for equality.

other The IParagraphStartIndentStyle to be tested against this, the current, one.

Getting and Setting the Indentation Value

Use these member functions to get and set the starting identation value.

setStartIndent

```
public:void setStartIndent(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the amount of indentation to be applied to the beginning of a paragraph.

value The amount of space in points, expressed as a float.

startIndent

```
public:float startIndent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current setting for indentation at the beginning of a paragraph.

IParagraphStartIndentStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

IParagraphStartIndentStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	465	getValueFromSet	465
isContainedInSet	465	isValueContainedInSet	465
removeFromSet	465		

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

IParagraphStyle

The IParagraphStyle class is an abstract base class for all paragraph-based styles, for example, those that can only apply to whole paragraphs.

Do not subclass any of the classes defined here or in istyles.hpp. The styles are designed to be a closed set.

Class header file: itxtstyl.hpp

IParagraphStyle - Hierarchy List

ITextStyle

IParagraphStyle

IParagraphSpreadStyle

IParagraphStartIndentStyle

ITabRulerStyle

IParagraphEndIndentStyle

IParagraphFirstIndentStyle

IParagraphJustificationStyle

IParagraphStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IParagraphStyle objects.

~IParagraphStyle

```
public:virtual ~IParagraphStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IParagraphStyle object.

IParagraphStyle

Constructs an IParagraphStyle object.

Overload 1

```
protected:IParagraphStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 2

```
protected:IParagraphStyle(const IParagraphStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an IParagraphStyle object that is copied from the given one.

other The IParagraphStyle object to be copied.

Assignment Operator

Use this operator to assign the given IParagraphStyle object to the current one.

operator =

```
protected:IParagraphStyle& operator =(const IParagraphStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The IParagraphStyle object to assign.

Obtaining the Style's Propagation Type

Use this member function to get the style's propagation type.

propagates

```
public:virtual ITextStyle::EStylePropagation propagates() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the style's propagation type. For IParagraphStyle, the value is ITextStyle::kPropagateByParagraph.

IParagraphStyle - Inherited Member Functions and Data

Inherited Public Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

Inherited Protected Data

None

IStandardTabRuler

IStandardTabRuler is a concrete class that implements the IMTabRuler protocol. It provides a mechanism to help you manage your tab stops.

Do not derive from this class.

Class header file: imtabrl.hpp

IStandardTabRuler - Hierarchy List

IMTabRuler

IStandardTabRuler

IStandardTabRuler - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IStandardTabRuler objects.

~IStandardTabRuler

```
public:virtual ~IStandardTabRuler()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IStandardRulerTab object.

IStandardTabRuler

Constructs an IStandardTabRuler object.

Overload 1

```
public:IStandardTabRuler( ioc::isize_t tabCount,
                          const ITabStop* tabs,
                          GCoord autoSpacing )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IStandardTabRuler(GCoord autoSpacing)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a tab ruler conaining only auto tabs, autoSpacing from 0 to eternity. Use this constructor to create an IStandardTabRuler whose tab positions are initialized using the given automatic spacing metric.

Overload 3

```
public:IStandardTabRuler()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor to create an empty standard tab ruler.

Overload 4

```
public:IStandardTabRuler(const IMTabRuler& ruler)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor. Use this constructor to create a standard tab ruler that is based on the given IMTabRuler.

Copying the Tab Ruler

Use this member function to clone a tab ruler.

clone

```
public:virtual IMTabRuler* clone() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a new tab ruler that is identical to this one.

Modifying Tab Stops

Use these member functions to move, add, or remove tab stops to the ruler.

addTab

```
public:virtual void addTab(const ITabStop& tabToAdd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds a new tab stop to the ruler. If the new tab stop has the same position as an existing tab stop, the existing one is removed.

moveTab

```
public:virtual void
  moveTab( GCoord fromPosition,
           GCoord toPosition )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves a tab stop to a new position. If the ruler contains a tab stop at fromPosition, moves it to toPosition; otherwise, does nothing.

removeTab

```
public:virtual void removeTab(GCoord position)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes a tab stop from the ruler. If the ruler contains a tab stop at position, it is removed; otherwise, nothing happens.

Obtaining a Copy of a Tab Stop

Use these member functions to copy the specified tab stop.

firstTab

```
public:virtual ITabStop firstTab() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the first tab in the ruler. If an autoTab, it is at position zero, and all subsequent tabs will be autotabs at autoSpacing intervals.

nextTab

```
public:virtual ITabStop nextTab(GCoord position) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the first tab in the ruler with fPosition > position. If it is an autotab, it is at an even multiple of autoSpacing, and all subsequent tabs will be autotabs at autoSpacing intervals.

Obtaining a Reference to a Tab Stop

Use these operators to obtain a constant or mutable reference to a tab stop.

operator []

Returns a reference to a tab stop.

Overload 1

```
public:virtual const ITabStop& operator [](ioc::isize_t ndx) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a const reference to the ″ndx″-th tab stop in the ruler.

Overload 2

```
public:virtual ITabStop& operator [](ioc::isize_t ndx)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a modifiable reference to the ″ndx″-th tab stop in the ruler.

Obtaining Information about Tab Stops

Use these member functions to determine the number of tab stops, the space interval between stop, or if the ruler contains a tab stop that matches the given tab stop position and type.

autoSpacing

```
public:virtual GCoord autoSpacing() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the interval for autotabs.

contains

```
public:virtual bool contains(const ITabStop& tabToTest) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the ruler contains a tab stop with the same position and type as tabStop.

tabCount

```
public:virtual ioc::isize_t tabCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of tab stops in this ruler.

IStandardTabRuler - Inherited Member Functions and Data

Inherited Public Functions

IMTabRuler			
Definition	**Page Number**	**Definition**	**Page Number**
autoSpacing	452	clone	451
firstTab	451	nextTab	452
operator ==	451	operator []	452
tabCount	452		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IString

The IString class provides arrays of characters. These objects are functionally equivalent to objects of the class I0String with one major distinction: IStrings are indexed starting at 1 instead of 0. (Note that if you use an index of zero as a starting position in an IString member function, this index value is automatically converted to an index of 1.)

IString provides an operator char*. To access the actual string contained in an object of type IString, cast the assignment variable implicitly or explicitly. IString is null-terminated so that it can be cast to a char* safely. However, the terminating null is not counted in its length, and the IString can still have embedded nulls. When cast to a char*, the string is truncated at the first null encountered.

IString member functions provide the following functions beyond that available from the standard C char* arrays and the STRING.H library functions:

- No restrictions on string contents. Thus, strings can contain NULL characters. **Note:** IString is null-terminated, so that it can be correctly cast to char*. This null is not counted in its length. When an IString is cast to a char*, the string is truncated at the first null encountered.
- Automatic conversion from and to numeric types.

- Automatic deletion of the string buffer when the IString is destroyed.
- Full support for the following:
  - All comparison operators.
  - All bitwise operators.
  - Concatenation using the more natural + operator.
- String data testing, such as for characters, digits, and hexadecimal digits.
- A full complement of the following:
  - String manipulation functions, such as center, left- and right-justification, stripping of leading and trailing characters, deleting substrings, and inserting strings.
  - Corresponding string manipulation functions that return a new IString rather than modifying the receiver.
  - String-searching functions, such as for the byte index of a string and the last-byte index of string.
- Word manipulation, such as the index of a word and the search for a word phrase.
- Support for mixed strings that contain both single-byte character set (SBCS) characters and double-byte character set (DBCS) characters.

When a program using IString objects is run on a DBCS system, the IString objects support DBCS characters within the string contents. The various IString search functions do not accidentally match an SBCS character with the second byte of a DBCS character that has the same value. Also, IString functions that modify IString objects, such as IString::subString, IString::remove, and IString::translate, never separate the two bytes of a DBCS character. If one of the two bytes of a DBCS character is removed, the remaining byte is replaced with the appropriate pad character (if the function performing the change has one) or a blank.

When working with IString objects that contain DBCS data, ensure that the contents are not altered in such a way as to corrupt the data. For example, the statement:

```
aString[n] = 'x';
```

would be in error if the nth character of the IString was the first or second byte of a DBCS character.

For any IString method that requires an index position, if the specified position is in the middle of a DBCS character, this 'partial' character will be ignored.

Note: Any function that reallocates an IString can throw an exception for out-of-range errors. These occur if you attempt to construct an IString with a length greater than that supported by ulength_t.

IString objects are held in IBuffer objects, which allocate the area for the character arrays using the C++ operator new. The only limitation for the size of an IString are the limitations imposed by the operating system.

IString does not provide methods that counts the number of logical DBCS characters or accessor to a DBCS character at a particular index. The reason is that they can be achieved through the next() method. For example:

To get the number of logical characters in a DBCS string:

```
IString str("some DBCS string");
for (const char* i = str, int j = 0; *i != NULL; i = str.next(i), j++);
// j now contains the number of logical DBCS characters.
```

AIX Considerations

DBCS is equivalent to multiple-byte character set (MBCS).

OS/2 Considerations

To obtain locale-sensitive result for many IString operations, such as comparison, searching, editing, etc., you must enable locale-based string operation by calling IString::enableInternationalization().

Windows Considerations

To obtain locale-sensitive result for many IString operations, such as comparison, searching, editing, etc., you must enable locale-based string operation by calling IString::enableInternationalization().

z/OS Considerations

DBCS is equivalent to multiple-byte character set (MBCS).

Class header file: istring.hpp

IString - Hierarchy List

IString

I0String

IString - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IString objects.

~IString

```
public:~IString()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IString object.

IString

You can construct objects of this class in the following ways:

- Construct a NULL string.
- Construct a string with the ASCII representation of a given numeric value, supporting all variations of integer and double.
- Construct a string with a copy of the specified character data, supporting ASCII strings, characters, and IStrings. The character data passed is converted to its ASCII representation.
- Construct a string with contents that consist of copies of up to three buffers of arbitrary data (void*). Optionally, you only need to provide the length, in which case the IString contents are initialized to a specified pad character. The default pad character is a blank.

These constructors can throw exceptions under the following conditions:

- Memory allocation errors

  Many factors dynamically allocate space and these allocation requests may fail. If so, the Open Class Library translates memory allocation

errors into exceptions. Generally, such errors do not occur until you allocate an astronomical amount of storage.

- Out-of-range errors.

  These occur if you attempt to construct an IString with a length greater than that supported by ulength_t.

Overload 1

```
public:IString(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of a long numeric value.

Overload 2

```
public:IString(unsigned anUnsigned)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of an unsigned numeric value.

Overload 3

```
public:IString(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified IString.

Overload 4

```
public:IString(unsigned short anUnsignedShort)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of an unsigned short numeric value.

Overload 5

```
public:IString(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of a double numeric value.

Overload 6

```
public:IString(char aChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the character. The string length is set to 1.

Overload 7

```
public:IString(unsigned char anUnsignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the unsigned character. The string length is set to 1.

Overload 8

```
public:IString(const char* pChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified ASCII string.

Overload 9

```
public:IString(signed char aSignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the signed character. The string length is set to 1.

Overload 10

```
public:IString(unsigned long anUnsignedLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of an unsigned long numeric value.

Overload 11

```
public:IString(const unsigned char* pUnsignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified unsigned ASCII string.

Overload 12

```
public:IString(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

Constructs a string representation of a long long value.

This constructor is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 13

```
public:IString(unsigned long long anUnsignedLongLong)
```

This is supported on **AIX OS/400 z/OS**

Constructs a string representation of an unsigned long long object.

This constructor is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 14

```
public:IString(const IString* pIString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a string from a const string pointer.

Overload 15

```
public:IString(const IText& anIText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified IText, using a transcoder based on the current locale (AIX and OS/2) or code page (Windows).

Overload 16

```
public:IString( const void* pBuffer1,
         ulength_t lenBuffer1,
         const void* pBuffer2,
         ulength_t lenBuffer2,
         const void* pBuffer3,
         ulength_t lenBuffer3,
         char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 17

```
public:IString( const void* pBuffer1,
         ulength_t lenBuffer1,
         char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 18

```
public:IString( const void* pBuffer1,
         ulength_t lenBuffer1,
         const void* pBuffer2,
         ulength_t lenBuffer2,
         char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 19

```
public:IString(const signed char* pSignedChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string with a copy of the specified signed ASCII string.

Overload 20

```
public:IString(short aShort)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of a short numeric value.

Overload 21

```
public:IString(int anInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a string representation of an integer value.

Overload 22

```
public:IString()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a NULL string.

Binary Conversions

These member functions work if isBinaryDigits() == true; if not, they return a null string. The static members by the same name can be applied to a string to return the modified string without changing the argument string.

b2c

Performs a two step conversion. First, the binary digits are converted to decimal numbers which are then stored in the result IString as raw binary data. For example, 110011_2 will be converted to 51_{10} which when stored as raw binary data represents the ASCII character '3'. Likewise, 110011 00110011_2 will result in a string ″33″.

Note: This function is not locale-sensitive.

Overload 1

```
public:IString& b2c()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

Overload 2

```
public:static IString b2c(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

b2d

Converts a string of binary digits to a string of decimal digits. For example, this function changes 00011001 to 25 and 0001001000110100 to 4660.

Overload 1

```
public:static IString b2d(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& b2d()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

b2x

Converts a string of binary digits to a string of hexadecimal digits. For example, this function changes 00011011 to 1b and 10001001000110100 to 11234.

Overload 1

```
public:static IString b2x(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& b2x()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Bit Operations

Use these member functions to implement various public members of this class requiring bitwise operations.

applyBitOp

```
protected:IString&
  applyBitOp( const char* pArg,
              ulength_t argLen,
              BitOperator op )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implements the bitwise operators &, |, and ^.

Character Conversions

These member functions convert a string to binary, numeric, or hexadecimal representation. The static members by the same name can be applied to a string to return the modified string without changing the argument string. These members are used much like the similar REXX functions. For example:

```
aString.c2b();                                  // Changes aString.
String binaryDigits = IString::c2b( aString ); // Leaves aString alone.
```

c2b

Converts a string as if it contains raw binary data to a string of 0 or 1 digits. For example, the character '3', which in ASCII is 51_{10} will be converted to the string "00110011" (which is 51 in decimal).

Note: This function is not locale-sensitive.

Overload 1

```
public:IString& c2b()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
z/OS Considerations

Mappings are based on code page IBM-1047 only.

Overload 2

```
public:static IString c2b(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
z/OS Considerations

Mappings are based on code page IBM-1047 only.

c2d

Converts a string as if it contains raw binary data to a string of decimal digits. For example, the character '3', which in ASCII is 51_{10} will be converted to the string "51".

Note: This function is not locale-sensitive.

Overload 1

```
public:static IString c2d(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& c2d()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

c2x

Converts a string as if it contains raw binary data to a string of hexadecimal digits. For example, the character '3', which in ASCII is 51_{10} will be converted to the string "33" (which is 51 in decimal).

Note: This function is not locale-sensitive.

Overload 1

```
public:static IString c2x(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

Overload 2

```
public:IString& c2x()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

Counting

Use these member functions to determine the number of the specified type.

occurrencesOf

Returns the number of occurrences of the specified IString, char*, char, or IStringTest. If you just want a boolean test, this is slower than IString::indexOf.

Overload 1

```
public:ucount_t
  occurrencesOf( char aCharacter,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ucount_t
  occurrencesOf( const IString& aString,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ucount_t
  occurrencesOf( const char* pString,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:ucount_t
  occurrencesOf( const IStringTest& aTest,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
protected:ucount_t
  occurrencesOf( const char* pSearchString,
                 ulength_t searchLen,
                 uoffset_t startPos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Diagnostics

These member funcitons provide IString diagnostic information for IString objects. Often, you use these members to write trace information when debugging.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns information about the IString's internal representation that you can use for debugging.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the string itself, so that IString supports this common protocol supported by other classes in the Open Class Library.

Editing

Use these member functions to edit a string. All return a reference to the modified receiver. Many that are length-related, such as center and leftJustify, accept a pad character that defaults to a blank. In all cases, you can specify argument strings as either objects of the IString class or by using char*.

Static members by the same name can be applied to an IString to obtain the modified IString without affecting the argument. For example:

```
aString.change('\t', '   ');                  // Changes all tabs in aString to 3 blanks.
IString s = IString::change( aString, '\t', '   ' );  // Leaves aString as is.
```

center

Centers the receiver within a string of the specified length.

Overload 1

```
public:IString&
  center( ulength_t length,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  center( const IString& aString,
          ulength_t length,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

change

Changes occurrences of a specified pattern to a specified replacement string. You can specify the number of changes to perform. The default is to change all occurrences of the pattern. You can also specify the position in the receiver at which to begin.

The parameters are the following:

inputString
The pattern string as a reference to an object of type IString. The library searches for the pattern string within the receiver's data.

pInputString
The pattern string as a NULL-terminated string. The library searches for the pattern string within the receiver's data.

outputString
The replacement string as a reference to an object of type IString. It replaces the occurrences of the pattern string in the receiver's data.

pOutputString
The replacement string as a NULL-terminated string. It replaces the occurrences of the pattern string in the receiver's data.

startPos
The position to start the search at within the receiver's data. The default is 1.

numChanges
The number of patterns to search for and change. The default is IString::kAllChanges, which causes changes to all occurrences of the pattern.

Overload 1

```
public:IString&
  change( const IString& inputString,
          const IString& outputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  change( const IString& aString,
          const IString& inputString,
          const IString& outputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString&
  change( const char* pInputString,
          const IString& outputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  change( const IString& aString,
          const char* pInputString,
          const IString& outputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IString&
  change( const IString& inputString,
          const char* pOutputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:static IString
  change( const IString& aString,
          const char* pInputString,
          const char* pOutputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:static IString
  change( const IString& aString,
          const IString& inputString,
          const char* pOutputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IString&
  change( const char* pInputString,
          const char* pOutputString,
          uoffset_t startPos = 1,
          ucount_t numChanges = kAllChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
protected:IString&
  change( const char* pPattern,
          ulength_t patternLen,
          const char* pReplacement,
          ulength_t replacementLen,
          uoffset_t startPos,
          ucount_t numChanges )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

copy

Replaces the receiver's contents with a specified number of replications of itself.

Overload 1

```
public:static IString
  copy( const IString& aString,
        ucount_t numCopies )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& copy(ucount_t numCopies)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

insert

Inserts the specified string after the specified location.

Overload 1

```
public:IString&
  insert( const char* pString,
          uoffset_t index = 0,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  insert( const IString& aString,
          const char* pInsert,
          uoffset_t index = 0,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:static IString
  insert( const IString& aString,
          const IString& anInsert,
          uoffset_t index = 0,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IString&
  insert( const IString& aString,
          uoffset_t index = 0,
          char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
protected:IString&
  insert( const char* pInsert,
          ulength_t insertLen,
          uoffset_t startPos,
          char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

leftJustify

Left-justifies the receiver in a string of the specified length. If the new length, **length**, is larger than the current length, the string is extended by the pad character, **padCharacter**. The default pad character is a blank.

Overload 1

```
public:IString&
  leftJustify( ulength_t length,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  leftJustify( const IString& aString,
               ulength_t length,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lowerCase

Translates all uppercase letters in the receiver to lowercase.

Overload 1

```
public:static IString lowerCase(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& lowerCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

overlayWith

Replaces a specified portion of the receiver's contents with the specified string. The overlay starts in the receiver's data at the **index**, which defaults to 1. If **index** is beyond the end of the receiver's data, it is padded with the pad character, **padCharacter**.

Overload 1

```
public:IString&
  overlayWith( const char* pString,
               uoffset_t index = 1,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  overlayWith( const IString& aString,
               const IString& anOverlay,
               uoffset_t index = 1,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString&
  overlayWith( const IString& aString,
               uoffset_t index = 1,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  overlayWith( const IString& aString,
               const char* pOverlay,
               uoffset_t index = 1,
               char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
protected:IString&
  overlayWith( const char* pOverlay,
               ulength_t overlayLen,
               uoffset_t index,
               char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

remove

Deletes the specified portion of the string (that is, the substring) from the receiver. You can use this function to truncate an IString object at a specific position. For example:

```
aString.remove(8);
```

removes the substring beginning at index 8 and takes the rest of the string as a default.

Overload 1

```
public:IString& remove(uoffset_t startPos, ucount_t numChars)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  remove( const IString& aString,
          uoffset_t startPos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString& remove(uoffset_t startPos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  remove( const IString& aString,
          uoffset_t startPos,
          ucount_t numChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

reverse

Reverses the receiver's contents.

Overload 1

```
public:static IString reverse(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& reverse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

rightJustify

Right-justifies the receiver in a string of the specified length. If the receiver's data is shorter than the requested length, **length**, it is padded on the left with the pad character, **padCharacter**. The default pad character is a blank.

Overload 1

```
public:IString&
  rightJustify( ulength_t length,
                char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  rightJustify( const IString& aString,
                ulength_t length,
                char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

space

```
public:IString&
  space( ucount_t numSpaces = 1,
         char spaceChar = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Modifies the receiver so that all words are separated by the specified number of blanks. The default is one blank. All white space is converted to simple blanks.

Note: The static functions IString::space and IString::removeWords obtain the same result but do not affect the String to which they are applied.

strip

Strips both the leading and trailing character or characters. You can specify the character or characters as the following:

- A single char
- A char* array
- An IString object
- An IStringTest object

The default is white space.

Overload 1

```
public:static IString
  strip( const IString& aString,
         const char* pStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  strip( const IString& aString,
         const IStringTest& aTest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString& strip()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IString& strip(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IString& strip(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IString& strip(const IStringTest& aTest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:static IString
  strip( const IString& aString,
         const IString& aStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IString& strip(char aCharacter)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:static IString strip(const IString& aString, char aChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 10

```
protected:IString&
  strip( const char* pChar,
         ulength_t len,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 11

```
protected:IString&
  strip( const IStringTest& aTest,
         IStringEnum::StripMode mode )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

stripBlanks

```
public:static IString stripBlanks(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Strips both leading and trailing white space.

Note: This function is the static version of IString::strip, which has been renamed to avoid a duplicate definition.

stripLeading

Strips the leading character or characters.

Overload 1

```
public:IString& stripLeading(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& stripLeading(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString& stripLeading(const IStringTest& aTest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  stripLeading( const IString& aString,
                char aChar )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:static IString
  stripLeading( const IString& aString,
                const char* pStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:static IString
  stripLeading( const IString& aString,
                const IString& aStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:static IString
  stripLeading( const IString& aString,
                const IStringTest& aTest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IString& stripLeading()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:IString& stripLeading(char aCharacter)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

stripLeadingBlanks

```
public:static IString stripLeadingBlanks(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Strips the leading character or characters.

Note: This function is the static version of IString::stripLeading, which has been renamed to avoid a duplicate definition.

stripTrailing

Strips the trailing character or characters.

Overload 1

```
public:static IString
  stripTrailing( const IString& aString,
                 const IStringTest& aTest )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  stripTrailing( const IString& aString,
                 char aChar )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString& stripTrailing()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IString& stripTrailing(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IString& stripTrailing(char aCharacter)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IString& stripTrailing(const IStringTest& aTest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:static IString
  stripTrailing( const IString& aString,
                 const char* pStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:static IString
  stripTrailing( const IString& aString,
                 const IString& aStringOfChars )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:IString& stripTrailing(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

stripTrailingBlanks

```
public:static IString stripTrailingBlanks(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Strips the trailing character or characters.

Note: This function is the static version of IString::stripTrailing, which has been renamed to avoid a duplicate definition.

translate

Converts all of the receiver's characters that are in the first specified string to the corresponding character in the second specified string.

Overload 1

```
public:static IString
  translate( const IString& aString,
             const IString& inputChars,
             const IString& outputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString&
  translate( const char* pInputChars,
             const char* pOutputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString&
  translate( const char* pInputChars,
             const IString& outputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  translate( const IString& aString,
             const char* pInputChars,
             const char* pOutputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:static IString
  translate( const IString& aString,
             const char* pInputChars,
             const IString& outputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:static IString
  translate( const IString& aString,
             const IString& inputChars,
             const char* pOutputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:IString&
  translate( const IString& inputChars,
             const char* pOutputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IString&
  translate( const IString& inputChars,
             const IString& outputChars,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
protected:IString&
  translate( const char* pInputChars,
             ulength_t inputLen,
             const char* pOutputChars,
             ulength_t outputLen,
             char padCharacter )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

upperCase

Translates all lowercase letters in the receiver to uppercase.

Overload 1

```
public:IString& upperCase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString upperCase(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Forward Searches

These member functions support searching a string in various ways. You can specify an optional index that indicates the search start position. The default starts at the beginning of the string.

indexOf

Returns the byte index of the first occurrence of the specified string within the receiver. If there are no occurrences, 0 is returned. In addition to IStrings, you can also specify a single character or an IStringTest.

Overload 1

```
public:uoffset_t
  indexOf( char aCharacter,
           uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOf( const char* pString,
           uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOf( const IString& aString,
           uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOf( const IStringTest& aTest,
           uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyBut

Returns the index of the first character of the receiver that is not in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that fails the test prescribed by a specified IStringTest object.

Overload 1

```
public:uoffset_t
  indexOfAnyBut( char validChar,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOfAnyBut( const char* pValidChars,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOfAnyBut( const IStringTest& aTest,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOfAnyBut( const IString& validChars,
                 uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfAnyOf

Returns the index of the first character of the receiver that is a character in the specified set of characters. If there are no characters, 0 is returned. Alternatively, this function returns the index of the first character that passes the test prescribed by a specified IStringTest object.

Overload 1

```
public:uoffset_t
  indexOfAnyOf( char searchChar,
                uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  indexOfAnyOf( const IString& searchChars,
                uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  indexOfAnyOf( const char* pSearchChars,
                uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  indexOfAnyOf( const IStringTest& aTest,
                uoffset_t startPos = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

indexOfPhrase

```
public:uoffset_t
  indexOfPhrase( const IString& wordString,
                 ucount_t startWord = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the position of the first occurrence of the specified phrase in the receiver. If the phrase is not found, 0 is returned.

indexOfWord

```
protected:uoffset_t
  indexOfWord( ucount_t wordNumber,
               uoffset_t startPos,
               ucount_t numWords ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the index of the specified white-space-delimited word in the receiver. If the word is not found, 0 is returned.

Hex Conversions

These member functions work if isHexDigits() == true; if not, they return a NULL string. The static members by the same name can be applied to a string to return the modified string without changing the argument string.

x2b

Converts a string of hexadecimal digits to a string of binary digits. For example, this function changes a1c to 101000011100 and f3 to 11110011.

Overload 1

```
public:IString& x2b()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString x2b(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

x2c

Converts a string of hexadecimal digits to a string of raw data. This is the reverse process of c2x(). Please check the documentation for c2x() for details.

Note: This function is not locale-sensitive.

Overload 1

```
public:static IString x2c(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
z/OS Considerations

Mappings are based on code page IBM-1047 only.

Overload 2

```
public:IString& x2c()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
z/OS Considerations

Mappings are based on code page IBM-1047 only.

x2d

Converts a string of hexadecimal digits to a string of decimal digits. For example, this function changes a1c to 2588 and 10000 to 65536.

Overload 1

```
public:static IString x2d(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& x2d()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

These member functions are used to implement this class; specifically, they initialize or set the underlying IBuffer data.

initBuffer

Resets the contents from a specified buffer or buffers.

Overload 1

```
protected:IString& initBuffer(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IString& initBuffer(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IString& initBuffer(unsigned long long anUnsignedLongLong)
```

This is supported on **AIX OS/400 z/OS**

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 4

```
protected:IString&
  initBuffer( const void* p1,
              ulength_t len1,
              const void* p2 = 0,
              ulength_t len2 = 0,
              const void* p3 = 0,
              ulength_t len3 = 0,
              char padChar = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
protected:IString& initBuffer(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 6

```
protected:IString& initBuffer(unsigned long anUnsignedLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

setBuffer

```
protected:IString& setBuffer(IBuffer* ibuff)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the private data member to point to a new IBuffer object.

Manipulation

Use these member functions to manipulate a string's contents. All are overloaded so that standard C strings can be used efficiently without constructing an equivalent string first.

operator &

Performs bitwise AND. This function can handle the following three forms:

string1 & aString
Both operands are of type IString.

string1 & pString
The first operand is an IString and the second is a NULL-terminated character string.

pString & aString
The first operand is a NULL-terminated character string and the second is an IString.

Overload 1

```
public:IString operator &(const char* pString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString operator &(const IString& aString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &=

Performs bitwise AND and replaces the receiver. This function can handle the following two forms:

string1 &= aString
Both operands are of type IString.

string1 &= pString
The first operand is an IString and the second is a NULL-terminated character string.

Overload 1

```
public:IString& operator &=(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& operator &=(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

Concatenates two strings. This function can handle the following three forms:

string1 + aString
Both operands are of type IString.

string1 + pString
The first operand is an IString and the second is a NULL-terminated character string.

pString + aString
The first operand is a NULL-terminated character string and the second is an IString.

Overload 1

```
public:IString operator +(const char* pString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString operator +(const IString& aString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +=

Concatenates the specified string to the receiver and replaces the receiver. This function can handle the following two forms:

string1 += aString
Both operands are of type IString.

string1 += pString
The first operand is an IString and the second is a NULL-terminated character string.

Overload 1

```
public:IString& operator +=(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& operator +=(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator =

```
public:IString& operator =(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the contents of the string.

operator ^

Performs bitwise XOR. This function can handle the following three forms:

string1 ^ aString
Both operands are of type IString.

string1 ^ pString
The first operand is an IString and the second is a NULL-terminated character string.

pString ^ aString
The first operand is a NULL-terminated character string and the second is an IString.

Overload 1

```
public:IString operator ^(const char* pString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString operator ^(const IString& aString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^=

Performs bitwise XOR and replaces the receiver. This function can handle the following two forms:

string1 ^= aString
Both operands are of type IString.

string1 ^= pString
The first operand is an IString and the second is a NULL-terminated character string.

Overload 1

```
public:IString& operator ^=(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& operator ^=(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator |

Performs bitwise OR. This function can handle the following three forms:

string1 | aString
Both operands are of type IString.

string1 | pString
The first operand is an IString and the second is a NULL-terminated character string.

pString | aString
The first operand is a NULL-terminated character string and the second is an IString.

Overload 1

```
public:IString operator |(const IString& aString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString operator |(const char* pString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator |=

Performs bitwise OR and replaces the receiver with the resulting string. This function can handle the following two forms:

string1 |= aString
Both operands are of type IString.

string1 |= pString
The first operand is an IString and the second is a NULL-terminated character string.

Overload 1

```
public:IString& operator |=(const char* pString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& operator |=(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ~

```
public:IString operator ~() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the string's bitwise negation (the string's complement).

NLS Testing

Use these member functions to test the characters that comprise a string. Basically, you use these members to determine if an IString contains only characters from a specific NLS character set (SBCS, MBCS, DBCS). Note that this testing is based on the number of bytes used to represent the character in memory, not their display width. For example, an SBCS character uses one byte of storage.

disableInternationalization

```
public:static void disableInternationalization()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disables locale-based string operations.

enableInternationalization

```
public:static void enableInternationalization(bool enable = true)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enables locale-based string operations.

includesDBCS

```
public:bool includesDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are DBCS (double-byte character set), true is returned.

Note: This function is interchangeable with includesMBCS.

includesMBCS

```
public:bool includesMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are MBCS (multiple-byte character set), true is returned.

Note: This function is interchangeable with includesDBCS.

includesSBCS

```
public:bool includesSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If any characters are SBCS (single-byte character set), true is returned.

isDBCS

```
public:bool isDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are DBCS, true is returned.

Note: This function is interchangeable with isMBCS.

isInternationalized

```
public:static bool isInternationalized()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines if locale-based string operation is in effect.

isMBCS

```
public:bool isMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are MBCS, true is returned.

Note: This function is interchangeable with isDBCS.

isSBCS

```
public:bool isSBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are SBCS, true is returned.

isValidDBCS

```
public:bool isValidDBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no DBCS characters have a second byte of 0, true is returned.

Note: This function is interchangeable with isValidMBCS.

isValidMBCS

```
public:bool isValidMBCS() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If no MBCS characters have a second byte of 0, true is returned.

Note: This function is interchangeable with isValidDBCS.

Numeric Conversions

These member functions work if isDigits() == true; if not, they return a NULL string. The static members by the same name can be applied to a string to return the modified string without changing the argument string.

d2b

Converts a string of decimal digits to a string of binary digits. This function builds the string eight bits at a time. For example,

```
'12' gets converted to '00001100'
'17' gets converted to '00010001'
'123' gets converted to '01111011'
```

Use stripLeading('0') to strip the leading zeros.

Overload 1

```
public:IString& d2b()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString d2b(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

d2c

Converts a string of decimal digits to a string of raw binary data. This is the reverse process of c2d(). Please check the method c2d() for details.

Note: This function is not locale-sensitive.

Overload 1

```
public:static IString d2c(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

Overload 2

```
public:IString& d2c()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

z/OS Considerations

Mappings are based on code page IBM-1047 only.

d2x

Converts a string of decimal digits to a string of hexadecimal digits. For example, this function changes 12 to c and 123 to 7b.

Overload 1

```
public:static IString d2x(const IString& aString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& d2x()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Pattern Matching

Use these member functions to determine if an object of this class contains a given pattern of characters.

includes

If the receiver contains the specified search string, true is returned.

Overload 1

```
public:bool includes(char aChar) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool includes(const IStringTest& aTest) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:bool includes(const char* pString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:bool includes(const IString& aString) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isAbbreviationFor

If the receiver is a valid abbreviation of the specified string, true is returned.

The parameters are the following:

fullString
The full string for the abbreviation check is contained in another IString.

pFullString
The full string for the abbreviation check is a NULL-terminated character string.

minAbbrevLength

The minimum length to match for it to be a valid abbreviation. The default minimum length is 0, which means the minimum length is the length of the receiver's string.

Overload 1

```
public:bool
  isAbbreviationFor( const IString& fullString,
                     ulength_t minAbbrevLength = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool
  isAbbreviationFor( const char* pFullString,
                     ulength_t minAbbrevLength = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isLike

If the receiver matches the specified pattern, which can contain global (or wildcard) characters, true is returned. (isLike does not guarantee to return correct result for comparison between SBCS and DBCS characters.)

- You can use the first global character to specify that 0 or more arbitrary characters are accepted. The default global character that does this is *, but you can specify another character when calling IString::isLike. For example:

  ```
  IString( "Allison" ).isLike( "Al*ison" ) -> true
  ```

- You can use the second global character to specify that a single arbitrary character is accepted. The default global character that does this is ?, but you can specify another character when calling IString::isLike. For example:

  ```
  IString( "istring7.cpp" ).isLike( "i*.?pp" ) -> true
  IString( "Not a question!" ).isLike( "*?", '*', '-' ) -> false
  ```

Overload 1

```
public:bool
  isLike( const IString& aPattern,
          char zeroOrMore = * IOCMSG ( ioc::kASTERISK ),
          char anyChar = * IOCMSG ( ioc::kQUESTION_MARK ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool
  isLike( const char* pPattern,
          char zeroOrMore = * IOCMSG ( ioc::kASTERISK ),
          char anyChar = * IOCMSG ( ioc::kQUESTION_MARK ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:bool
  isLike( const char* pPattern,
          ulength_t patternLen,
          char zeroOrMore,
          char anyChar ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Queries

Use these member functions to access general information about the string.

charType

```
public:IStringEnum::CharType charType(uoffset_t index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the type of the character at the specified index.

length

```
public:ulength_t length() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the length (in number of bytes) of the string, not counting the terminating NULL character.

To get the number of logical characters in a DBCS string, you can use the next() method. For example:

```
IString str("some DBCS string");
for (const char* i = str, int j = 0; *i != NULL; i = str.next(i), j++);
// j now contains the number of logical DBCS characters.
```

operator []

Returns a reference to the specified byte of the string.

Note: If you call the non-const version of this function with an index beyond the end, the function extends the string.

To get to a particular logical character in a DBCS string, you can use the next() method. For example:

```
IString str("some DBCS string");
int index = 5; // move to the 5th DBCS character.
for (const char* i = str, int j = 0; *i != NULL && j != index; i = str.next(i), j++);
// i now points to the 5th DBCS character.
```

Overload 1

```
public:const char& operator [](unsigned long long index) const
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 2

```
public:const char& operator [](signed long long index) const
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 3

```
public:const char& operator [](unsigned index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:char& operator [](unsigned long long index)
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 5

```
public:const char& operator [](signed index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:char& operator [](unsigned index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:const char& operator [](signed long index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
Exception

Exception	Description
IInvalidRequest	Passed an index larger than the string size. Possible causes include boundary errors and using this function instead of the non-const version, which grows the underlying IString buffer to accommodate the index value.

Overload 8

```
public:const char& operator [](unsigned long index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:char& operator [](signed long long index)
```

This is supported on **AIX OS/400 z/OS**

This operator is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

Overload 10

```
public:char& operator [](signed long index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 11

```
public:char& operator [](signed index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 12

```
public:char& operator [](unsigned long index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

size

```
public:ulength_t size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the length (in number of bytes) of the string, not counting the terminating NULL character.

To get the number of logical characters in a DBCS string, you can use the next() method. For example:

```
IString str("some DBCS string");
for (const char* i = str, int j = 0; *i != NULL; i = str.next(i), j++);
// j now contains the number of logical DBCS characters.
```

buffer

```
protected:IBuffer* buffer() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the address of the IBuffer referred to by this IString.

data

```
protected:char* data() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the address of the contents of the IString.

defaultBuffer

```
protected:static char* defaultBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the contents of the nullBuffer data member.

lengthOf

```
protected:static ulength_t lengthOf(const char* pChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the length (in number of bytes) of a C character array.

nullBuf

```
protected:static const char* nullBuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the value of nullBuffer.

Reverse Searches

These member functions permit searching the string in various ways. The lastIndexOf versions correspond to forward search indexOf members but start the search from the end of the string. These members return the index of the last character in the receiver IString that satisfies the search criteria. Also, they accept an optional argument that specifies where the search is to begin. The default is to start searching at the end of the string. Searching proceeds from right to left for these members.

lastIndexOf

Returns the index of the last occurrence of the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The returned value is in the range 0 <= x <= **startPos**. The default of IString::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 1 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

If **startPos** is in the middle of a DBCS character, this 'partial' character will be ignored during the search.

Overload 1

```
public:uoffset_t
  lastIndexOf( char aCharacter,
               uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOf( const IStringTest& aTest,
               uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOf( const char* pString,
               uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOf( const IString& aString,
               uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyBut

Returns the index of the last character not in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of IString::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 1 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

If **startPos** is in the middle of a DBCS character, this 'partial' character will be ignored during the search.

Overload 1

```
public:uoffset_t
  lastIndexOfAnyBut( const IStringTest& aTest,
                     uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOfAnyBut( char validChar,
                     uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOfAnyBut( const char* pValidChars,
                     uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOfAnyBut( const IString& validChars,
                     uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

lastIndexOfAnyOf

Returns the index of the last character in the specified string or character. The search starts at the position specified by **startPos** (inclusive) and proceeds backward. The default of IString::kEndOfString starts the search at the end of the receiver's string. If the search target is not found, 0 is returned.

If you specify 1 for **startPos**, the search starts at the beginning of the string. Therefore, because the search proceeds backward from its starting position, in this case the search target must occur at the beginning of the string for it to be found.

Overload 1

```
public:uoffset_t
  lastIndexOfAnyOf( const IStringTest& aTest,
                    uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:uoffset_t
  lastIndexOfAnyOf( const char* pSearchChars,
                    uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:uoffset_t
  lastIndexOfAnyOf( const IString& searchChars,
                    uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:uoffset_t
  lastIndexOfAnyOf( char searchChar,
                    uoffset_t startPos = kEndOfString ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stream Input

Use these member functions to read IStrings from standard C++ streams.

lineFrom

Returns the next line from the specified input stream. This static function accepts an optional line delimiter, which defaults to \n. The resulting

IString contains the characters up to the next occurrence of the delimiter. The delimiter character is skipped. If an EOF condition occurs, this function returns an IString whose contents are NULL.

Overload 1

```
public:static IString
  lineFrom( std::istream& aStream,
            char delim = * IOCMSG ( ioc::kNEWLINE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static IString
  lineFrom( istream& aStream,
            char delim = * IOCMSG ( ioc::kNEWLINE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streaming Support

These member functions support binary and test mode streaming of strings.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this string object in from the passed data stream.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this string object out to the passed data stream.

readFromStream

```
protected:void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Support method for streaming in of string objects from data streams.

writeToStream

```
protected:void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Support method for streaming out of string objects to data streams.

Subset

Use these member functions when a subset of characters is required.

subString

Returns a specified portion of the string (that is, the substring) of the receiver.

The parameters are the following:

startPos

The starting position of the substring being extracted. If this position is beyond the end of the data in the receiver, this function returns a NULL IString.

length The length of the substring to be extracted. If the length extends beyond the end of the receiver's data, the returned IString is padded to the specified length with **padCharacter**. If you do not specify **length** and it defaults, this function uses the rest of the receiver's data starting from **startPos** for padding.

padCharacter
The character the function uses as padding if the requested length extends beyond the end of the receiver's data. The default **padCharacter** is a blank.

You can use this function to truncate an IString object at a specific position. For example:

```
aString = aString.subString(1, 7);
```

returns the substring concluding with index 7 and discards the rest of the string.

Overload 1

```
public:IString subString(uoffset_t startPos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString
  subString( uoffset_t startPos,
             ulength_t length,
             char padCharacter = * IOCMSG ( ioc::kSPACE ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Testing

Use these member functions to determine if an IString contains only characters from a predefined set.

isAlphabetic

```
public:bool isAlphabetic() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are defined as alphabetic characters in the current locale.

isAlphanumeric

```
public:bool isAlphanumeric() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are defined as alphabetic characters in the current locale or are within the character range '0' to '9'.

isASCII

```
public:bool isASCII() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {0x00-0x7F}, true is returned.

isBinaryDigits

```
public:bool isBinaryDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are either 0 or 1, true is returned.

isControl

```
public:bool isControl() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are control characters. Control characters are determined using the iscntrl() C Library function defined in the cntrl locale source file and in the cntrl class of the LC_CTYPE category of the current locale. For example, on ASCII operating systems, control characters are those in the range {0x00-0x1F,0x7F}.

isDigits

```
public:bool isDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9'}, true is returned.

isGraphics

```
public:bool isGraphics() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are graphics characters.

Graphics characters are printable characters excluding the space character, as defined by the isgraph() C Library function in the graph locale source file and in the graph class of the LC_CTYPE category of the current locale. On ASCII systems, for example, graphics characters are those in the range {0x21-0x7E}.

isHexDigits

```
public:bool isHexDigits() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'0'-'9','A'-'F','a'-'f'}, true is returned.

isLowerCase

```
public:bool isLowerCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'a'-'z'}, true is returned.

isPrintable

```
public:bool isPrintable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are printable characters. Printable characters are defined by the isprint() C Library function as defined in the print locale source file and in the print class of the LC_CTYPE category of the current locale. On ASCII systems, for example, printable characters are those in the range {0x20-0x7E}.

isPunctuation

```
public:bool isPunctuation() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If none of the characters is white space, a control character, or an alphanumeric character, true is returned.

isUpperCase

```
public:bool isUpperCase() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If all the characters are in {'A'-'Z'}, true is returned.

isWhiteSpace

```
public:bool isWhiteSpace() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the characters are white-space characters. White-space characters are defined by the isspace() C Library function as defined in the space locale source file and in the space class of the LC_CTYPE category of the current locale. For example, on ASCII systems, white-space characters are those in the range {0x09-0x0D,0x20}.

isAbbrevFor

```
protected:bool
  isAbbrevFor( const char* pFullString,
               ulength_t fullLen,
               ulength_t minLen ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the receiver is a valid abbreviation of the specified string, true is returned.

The parameters are the following:

pFullString
The full string for the abbreviation check. The string can be either a NULL-terminated character string or not.

fullLen
The full length of the specified **pFullString** minus the null terminator.

minLen
The minimum length to match for it to be a valid abbreviation. If you specify 0, the minimum length is the length of the receiver's string.

Type Conversions

Use these member functions to convert a string to various other data types. The types supported are the same set as are supported by the IString constructors.

asDouble

```
public:double asDouble() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns, as a double, the number that the string represents.

asInt

```
public:long asInt() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number that the string represents as a long integer.

Note: If an IString contains non-numeric characters, this function returns the integer for the portion of the IString up to, but not including, the non-numeric character. The rest of the IString, following the invalid character, is not returned.

If an IString is larger than the maximum integer, this function returns the maximum integer, not the larger value.

asLongLong

```
public:long long asLongLong() const
```

This is supported on **AIX OS/400 z/OS**

Returns the number that the string represents as a long long integer.

Note: If an IString contains nonnumeric characters, this function returns the integer for the portion of the IString up to, but not including, the nonnumeric character. The rest of the IString, following the invalid character, is not returned.

If an IString is larger than the maximum long long, this function returns the maximum long long, not the larger value.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

asUnsigned

```
public:unsigned long asUnsigned() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns, as an unsigned long, the integer that the string represents.

asUnsignedLongLong

```
public:unsigned long long asUnsignedLongLong() const
```

This is supported on **AIX OS/400 z/OS**

Returns, as an unsigned long long, the long long that the string represents.

This function is available if IC_LONG_LONG is defined for the platform. See icomdefs.h.

operator char *

```
public:operator char *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a char* pointer to the string's contents. IString is null-terminated so that it can be cast to a char* safely. However, the terminating null is not counted in its length, and the IString can still have embedded nulls. When cast to a char*, the string is truncated at the first null encountered.

operator signed char *

```
public:operator signed char *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a signed char* pointer to the string's contents. IString is null-terminated so that it can be cast to a char* safely. However, the terminating null is not counted in its length, and the IString can still have embedded nulls. When cast to a char*, the string is truncated at the first null encountered.

operator unsigned char *

```
public:operator unsigned char *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an unsigned char* pointer to the string's contents. IString is null-terminated so that it can be cast to a char* safely. However, the terminating null is not counted in its length, and the IString can still have embedded nulls. When cast to a char*, the string is truncated at the first null encountered.

Utility Data

These protected static data members provide useful values for implementing IString. IString uses the various representation of NULL and zero for initialization and comparison purposes.

maxLong

```
protected:static const char * const maxLong
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The maximum value of a long, with 32-bit unsigned long integers.

OS/2 Considerations

This value is ″2147483647″ on OS/2 with 32-bit unsigned long integers.

null

```
protected:static const char * const null
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A string that contains no element.

zero

```
protected:static const char * const zero
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The number 0.

Word Operations

These member functions operate on a string as a collection of words separated by white-space characters. They find, remove, and count words or phrases.

indexOfWord

```
public:uoffset_t indexOfWord(ucount_t wordNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the index of the specified white-space-delimited word in the receiver. If the word is not found, 0 is returned.

lengthOfWord

```
public:ulength_t lengthOfWord(ucount_t wordNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the length of the specified white-space-delimited word in the receiver.

numWords

```
public:ucount_t numWords() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of words in the receiver.

removeWords

Deletes the specified words from the receiver's contents. You can specify the words by using a starting word number and the number of words. The latter defaults to the rest of the string.

Note: The static functions IString::space and IString::removeWords obtain the same result but do not affect the String to which they are applied.

Overload 1

```
public:static IString
  removeWords( const IString& aString,
               ucount_t startWord )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString& removeWords(ucount_t firstWord, ucount_t numWords)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IString& removeWords(ucount_t firstWord)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:static IString
  removeWords( const IString& aString,
               ucount_t startWord,
               ucount_t numWords )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

word

```
public:IString word(ucount_t wordNumber) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the specified white-space-delimited word in the receiver.

wordIndexOfPhrase

```
public:ucount_t
  wordIndexOfPhrase( const IString& aPhrase,
                     ucount_t startWord = 1 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the word number of the first word in the receiver that matches the specified phrase. The function starts its search with the word number you specify in **startWord**, which defaults to 1. If the phrase is not found, 0 is returned.

words

Returns a substring of the receiver that starts at a specified word and is comprised of a specified number of words. The word separators are copied to the result intact.

Overload 1

```
public:IString words(ucount_t firstWord, ucount_t numWords) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IString words(ucount_t firstWord) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

findPhrase

```
protected:uoffset_t
  findPhrase( const IString& aPhrase,
              ucount_t startWord,
              IndexType charOrWord ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates a specified string of words for indexOfWord functions.

IString - Type Definitions

BitOperator

```
typedef enum { bitAnd , bitOr , bitExclusiveOr , and = bitAnd ,
               or = bitOr , exclusiveOr = bitExclusiveOr } BitOperator
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use these enumerators to specify the bit operator to apply to the applyBitOp function. Valid bit operators are as follows:

- bitAnd
- bitOr
- bitExclusiveOr
- and
- or
- exclusiveOr

and, or, exclusiveOr has the same value as bitAnd, bitOr, bitExclusiveOr and may be removed in a future release.

IndexType

```
typedef enum { charIndex , wordIndex } IndexType
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use these enumerators to specify whether the result from the findPhrase function is a word index or a character index:

charIndex
Returns the result as the byte index within the string

wordIndex

Returns the result as the index of the matching word. For example, the first word is 1, the second word is 2, and so forth.

Related Enumeration

BitOperator

offset_t

Defines offset_t to be different values depending on the platform and the application.

To create portable applications, use this type definition instead of the native types.

Variation 1

```
typedef signed long offset_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines offset_t to be a signed long for 64 bit AIX applications and a signed int for 32 bit AIX applications.

OS/2 Considerations

Defines offset_t to be a signed int for 32 bit applications.

OS/400 Considerations

Defines offset_t to be a signed long long for AS/400 applications.

Windows Considerations

Defines offset_t to be a signed int for 32 bit applications.

Variation 2

```
typedef signed long long offset_t
```

This is supported on **OS/400**

AIX Considerations

Defines offset_t to be a signed long for 64 bit AIX applications and a signed int for 32 bit AIX applications.

OS/2 Considerations

Defines offset_t to be a signed int for 32 bit applications.

OS/400 Considerations

Defines offset_t to be a signed long long for AS/400 applications.

Windows Considerations

Defines offset_t to be a signed int for 32 bit applications.

Variation 3

```
typedef signed int offset_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines offset_t to be a signed long for 64 bit AIX applications and a signed int for 32 bit AIX applications.

OS/2 Considerations

Defines offset_t to be a signed int for 32 bit applications.

OS/400 Considerations

Defines offset_t to be a signed long long for AS/400 applications.

Windows Considerations

Defines offset_t to be a signed int for 32 bit applications.

ucount_t

Defines ucount_t to be different values depending on the platform and the application.

To create portable applications, use this type definition instead of the native types.

Variation 1

```
typedef unsigned long ucount_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines ucount_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ucount_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

Variation 2

```
typedef unsigned int ucount_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines ucount_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ucount_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

Variation 3

```
typedef unsigned long long ucount_t
```

This is supported on **OS/400**

AIX Considerations

Defines ucount_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ucount_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ucount_t to be an unsigned int for 32 bit applications.

ulength_t

Defines ulength_t to be different values depending on the platform and the application.

To create portable applications, use this type definition instead of the native types.

Variation 1

```
typedef unsigned long long ulength_t
```

This is supported on **OS/400**

AIX Considerations

Defines ulength_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ulength_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

Variation 2

```
typedef unsigned int ulength_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines ulength_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ulength_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

Variation 3

```
typedef unsigned long ulength_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines ulength_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines ulength_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines ulength_t to be an unsigned int for 32 bit applications.

uoffset_t

Defines uoffset_t to be different values depending on the platform and the application.

To create portable applications, use this type definition instead of the native types.

Variation 1

```
typedef unsigned long long uoffset_t
```

This is supported on **OS/400**

AIX Considerations

Defines uoffset_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines uoffset_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

Variation 2

```
typedef unsigned int uoffset_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines uoffset_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines uoffset_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

Variation 3

```
typedef unsigned long uoffset_t
```

This is supported on **WIN OS/2 AIX z/OS**

AIX Considerations

Defines uoffset_t to be an unsigned long for 64 bit AIX applications and an unsigned int for 32 bit AIX applications.

OS/2 Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

OS/400 Considerations

Defines uoffset_t to be an unsigned long long for AS/400 applications.

Windows Considerations

Defines uoffset_t to be an unsigned int for 32 bit applications.

IString - Associated Globals

kAllChanges

```
static const IString::ucount_t kAllChanges
```

This static constant is used by the IString::change member functions to define the default for the the number of patterns to search for and change. kAllChanges causes changes to all occurrences of the pattern.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

kEndOfString

```
static const IString::uoffset_t kEndOfString
```

This static constant is used by the IString::lastIndexOf, IString::lastIndexOfAnyBut and IString::lastIndexOfAnyOf member functions to define the starting position of the search. kEndOfString starts the search at the end of the string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IString& string1,
               const IString& string2 )
```

If the strings are not bitwise identical, true is returned. This operator handles the following form of the comparison:

string1!=string2
Both operands are of type IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const char* pString1,
               const IString& string2 )
```

If the strings are not bitwise identical, true is returned. This operator handles the following form of the comparison:

pString1!=string2
The first operand is a null-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IString& string1,
               const char* pString2 )
```

If the strings are not bitwise identical, true is returned. This operator handles the following form of the comparison:

string1!=pString2
The first operand is an IString and the second is a NULL-terminated character string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool operator !=(const IString& string1, char* pString2)
```

If the strings are not bitwise identical, true is returned. This operator handles the following form of the comparison:

string1!=pString2
The first operand is an IString and the second is a NULL-terminated character string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
IString
  operator &( const char* pString,
              const IString& aString )
```

Performs bitwise AND. This operator handles the following form of the comparison:

pString & aString
The first operand is a NULL-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IString
  operator +( const char* pString,
              const IString& aString )
```

Concatenates a null-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IString& string1,
              const IString& string2 )
```

If the first string is less than the second, using the standard collating scheme (memcmp), true is returned. This operator handles the following form of the comparison:

string1 < string2
Both operands are of type IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool operator <(const IString& string1, char* pString2)
```

If the first string is less than the second, using the standard collating scheme (memcmp), true is returned. This operator handles the following form of the comparison:

string1 < pString2
The first operand is an IString and the second is a NULL-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const char* pString1,
              const IString& string2 )
```

If the first string is less than the second, using the standard collating scheme (memcmp), true is returned. This operator can handles the following form of the comparison:

pString1 < string2
The first operand is a null-terminated character string and the second is an IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters

and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IString& string1,
              const char* pString2 )
```

If the first string is less than the second, using the standard collating scheme (memcmp), true is returned. This operator handles the following form of the comparison:

string1 < pString2
The first operand is an IString and the second is a NULL-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IString& aString )
```

Streams an IString's contents to an output stream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IString& aString )
```

Streams an IString's contents to an output stream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const IString& string1,
               const char* pString2 )
```

Equivalent to (string1 < string2) || (string1 == string2). This operator handles the following form of the comparison:

string1 <= pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const char* pString1,
               const IString& string2 )
```

Equivalent to (string1 < string2) || (string1 == string2). This operator can handle the following form of the comparison:

pString1 <= string2
The first operand is a null-terminated character string and the second is an IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool operator <=(const IString& string1, char* pString2)
```

Equivalent to (string1 < string2) || (string1 == string2). This operator handles the following form of the comparison:

string1 <= pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const IString& string1,
               const IString& string2 )
```

Equivalent to (string1 < string2) || (string1 == string2). This operator handles the following form of the comparison:

string1 <= string2
Both operands are of type IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const char* pString1,
               const IString& string2 )
```

If the strings are bitwise identical, true is returned. This operator handles the following form of the comparison:

pString1 == string2
The first operand is a null-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IString& string1,
               const IString& string2 )
```

If the strings are bitwise identical, true is returned. This operator handles the following form of the comparison:

string1 == string2
Both operands are of type IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IString& string1,
               const char* pString2 )
```

If the strings are bitwise identical, true is returned. This operator handles the following form of the comparison:

string1 == pString2
The first operand is an IString and the second is a null-terminated character string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool operator ==(const IString& string1, char* pString2)
```

If the strings are bitwise identical, true is returned. This operator handles the following form of the comparison:

string1 == pString2
The first operand is an IString and the second is a null-terminated character string.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool operator >(const IString& string1, char* pString2)
```

Equivalent to !(string1 <= string2). This operator handles the following form of the comparison:

string1 > pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const char* pString1,
              const IString& string2 )
```

Equivalent to !(string1 <= string2). This operator handles the following form of the comparison:

pString1 > string2
The first operand is a NULL-terminated character string and the second is an IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const IString& string1,
              const char* pString2 )
```

Equivalent to !(string1 <= string2). This operator handles the following form of the comparison:

string1 > pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const IString& string1,
              const IString& string2 )
```

Equivalent to !(string1 <= string2). This operator handles the following form of the comparison:

string1 > string2
Both operands are of type IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool operator >=(const IString& string1, char* pString2)
```

Equivalent to !(string1 < string2). This operator handles the following form of the comparison:

string1 >= pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IString& string1,
               const char* pString2 )
```

Equivalent to !(string1 < string2). This operator handles the following form of the comparison:

string1 >= pString2
The first operand is an IString and the second is a null-terminated character string.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const char* pString1,
               const IString& string2 )
```

Equivalent to !(string1 < string2). This operator handles the following form of the comparison:

pString1 >= string2
The first operand is a null-terminated character string and the second is an IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IString& string1,
               const IString& string2 )
```

Equivalent to !(string1 < string2). This operator handles the following form of the comparison:

string1 >= string2
Both operands are of type IString.

Note: This operator is not locale-sensitive. Because it uses memcmp and not strcoll, it compares the binary values representing the characters and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream &
  operator >>( istream& aStream,
               IString& aString )
```

Streams the next white-space-delimited word from an input stream into an IString.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream &
  operator >>( std::istream& aStream,
               IString& aString )
```

Streams the next white-space-delimited word from an input stream into an IString.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
IString
  operator ^( const char* pString,
              const IString& aString )
```

Performs bitwise XOR. This operator handles the following form:

pString ^ aString
The first operand is a NULL-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator |

```
IString
  operator |( const char* pString,
              const IString& aString )
```

Performs bitwise OR. This operator handles the following form:

pString | aString
The first operand is a null-terminated character string and the second is an IString.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

space

```
static IString
  space( const IString& aString,
         ucount_t numSpaces = 1,
         char spaceChar = * IOCMSG ( ioc::kSPACE ) )
```

Modifies the receiver so that all words are separated by the specified number of blanks. The default is one blank. All white space is converted to simple blanks.

Note: The static functions IString::space and IString::removeWords obtain the same result but do not affect the String to which they are applied.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IString - Inherited Member Functions and Data

Inherited Public Functions

IString			
Definition	**Page Number**	**Definition**	**Page Number**
space	488		

Inherited Public Data

IString			
Definition	**Page Number**	**Definition**	**Page Number**
kAllChanges	522	kEndOfString	522

Inherited Protected Functions

None

Inherited Protected Data

None

IStringEnum

The IStringEnum class serves as a repository for enumeration types related to the IString class. The Open Class Library places these enumeration types here so they can easily be shared between code that implements the classes IString, IBuffer, and IDBCSBuffer.

Class header file: istrenum.hpp

IStringEnum - Hierarchy List

IStringEnum

IStringEnum - Member Functions and Data by Group

IStringEnum - Type Definitions

CharType

```
typedef enum { sbcs , dbcs1 = 1 , mbcs1 = 1 , dbcs2 = 2 ,
               mbcs2 = 2 , mbcs3 = 3 , mbcs4 = 4 } CharType
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use these enumerators to specify the various types of characters that comprise an IString:

sbcs The IString contains single-byte character set (SBCS) characters.

dbcs1 The IString contains the first byte of a double-byte character support (DBCS) character.

dbcs2 The IString contains the second byte of a double-byte character support (DBCS) character.

mbcs1 The IString contains the first byte of a multi-byte character support (MBCS) character.

mbcs2 The IString contains the second byte of a multi-byte character support (MBCS) character.

mbcs3 The IString contains the third byte of a multi-byte character support (MBCS) character.

mbcs4 The IString contains the fourth byte of a multi-byte character support (MBCS) character.

Note: Code for MBCS must be the same as the length of the corresponding characters.

StripMode

```
typedef enum { leading , trailing , both } StripMode
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this enumeration to define the mode of various functions that strip leading characters, trailing characters, or both from IStrings.

leading
Strips the leading characters.

trailing
Strips the trailing characters.

both Strips both the leading and trailing characters.

IStringEnum - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStringGenerator

The IStringGenerator class manages the translation of objects to their IString form. Use the IStringGenerator::stringFor member function to translate an object to its IString representation. Call it on an object-by-object basis. In the default case (no IStringGeneratorFn), the object should support the operator-> or be an Object*, where the Object class has an asString member function.

The IStringGenerator::stringFor member function processes as follows:

1. If no IStringGeneratorFn object has been set (the default), then it uses an object's asString member function for translation.
2. If an optional IStringGeneratorFn has been set, it uses the IStringGeneratorFn::stringFor member function.

Note: IStringGeneratorFn objects represent functions. Objects of the IStringGeneratorMemberFn and IStringGeneratorRefMemberFn classes represent member functions and differ in how they treat the object parameter in the stringFor member function.

Create and use IStringGenerator objects by following these steps:

- Create an IStringGeneratorFn derived object containing the function to be called to generate an IString.
- Create an IStringGenereator object initializing it with the IStringGeneratorFn derived object.
- Call the IStringGenerator::stringFor member function passing as the parameter the object you want a string generated for.

Note: IStringGeneratorFn, as an abstract base class, establishes the protocol for specifying a function to translate an object to an IString. Inherit from this class and implement the stringFor member function to define the calling convention and call the stored function.

Class header file: istrgen.hpp

IStringGenerator - Hierarchy List

IStringGenerator

IStringGenerator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class. The copy constructor and operator= ensure that the contained generator function objects are reference-counted properly.

~IStringGenerator

```
public:virtual ~IStringGenerator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStringGenerator

Use the constructor to create an instance of a string generator object.

Overload 1

```
public:IStringGenerator( const ICountedPointerTo
  < IStringGeneratorFn < Element >
  >& generatorFunction = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You create a string generator object using an optional IStringGeneratorFn.

generatorFunction

A pointer (ICountedPointerTo) to an existing IStringGeneratorFn subclass object. If it is specified, subsequent calls to stringFor will use this function to produce strings for the given object.

If you do not specify a generator function, calls to stringFor will return an empty string.

If you want the same behavior as was provided by IBase::asString() in previous releases of Open Class Library, then you should specify an ICountedPointerTo pointer to an instance of IStringGeneratorasString for this parameter.

Note: The pointer object (ICountedPointerTo) ensures the generator function is reference counted properly.

Overload 2

```
public:IStringGenerator( const IStringGenerator
  < Element >& stringGenerator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You can create a string generator object using another IStringGenerator. This ensures that the reference-counting of the IStringGeneratorFn object is done properly.

stringGenerator

A string generator.

operator =

```
public:IStringGenerator < Element >&
  operator =( const IStringGenerator < Element >& stringGenerator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign one string generator to another. This ensures that the reference-counting of the IStringGeneratorFn object is done properly.

stringGenerator

A string generator.

Generate String

Use this member function to translate an object to its IString form.

stringFor

```
public:IString stringFor(const Element& element) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obtains an IString version of the element. If a local IStringGeneratorFn has been set, then it is used. Otherwise, the element's asString member function generates the IString.

element

Reference to the object for which it generates the string.

Note: For the default case (no IStringGeneratorFn), the element should natively support operator-> or the element must be an Object*, where the Object class has an asString member function.

IStringGeneratorRefMemberFn and IStringGeneratorMemberFn (IStringGeneratorFn derived classes) contain member functions that take no parameters. They support invoking these member functions on Object* elements (element is a pointer) and concrete Object elements (element is not a pointer), respectively. If you need additional string generation support, create your own IStringGeneratorFn derived class.

IStringGenerator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStringGeneratorasString

The IStringGeneratorasString template class is an IStringGeneratorFn-derived class. It dispatches C++ member functions against an object to generate an IString. It's purpose is to provide the same string generation capability as IBase::asString(), which was to obtain the standard version of an object's contents. (IBase is an obsolete class in this release of Open Class Library. and IStringGenerator no longer derives from it.)

Note: This class is used primarily by the Visual Builder. You would not normally use this class yourself.

Class header file: istrgen.hpp

IStringGeneratorasString - Hierarchy List

IMRefCounted

IStringGeneratorFn

IStringGeneratorasString

IStringGeneratorasString - Member Functions and Data by Group

Constructors & Destructor

You would not normally create objects of this class.

~IStringGeneratorasString

```
public:virtual ~IStringGeneratorasString()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStringGeneratorasString

```
public:IStringGeneratorasString()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an object of this class.

Create an instance of this string generator function that generates a string representation for objects of class **T**.

Generate String

Establishes the protocol for generating an IString for the object parameter. Subclasses implement this function to generate an IString for that object.

stringFor

```
public:virtual IString stringFor(const T& element) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the appropriate member function on the object reference parameter. The returned string will be of the same form as returned by IBase::asString() in previous releases of Open Class Library.

element

Reference to the object against which it dispatches the member function.

IStringGeneratorasString - Inherited Member Functions and Data

Inherited Public Functions

IStringGeneratorFn			
Definition	**Page Number**	**Definition**	**Page Number**
~IStringGeneratorFn	538	IStringGeneratorFn	538
stringFor	538		

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef	57	count	58
removeRef	57		

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted	57	IMRefCounted	57
operator =	57		

Inherited Protected Data

None

IStringGeneratorFn

The IStringGeneratorFn template class is an abstract base class defining the protocol for storing and calling functions to generate IStrings. Objects of this class represent functions to be called when the stringFor function is called. The stringFor pure virtual function accepts an object reference of the template class type.

Note: IStringGenerator does the following:

- Uses these objects (calls IStringGeneratorFn::stringFor) to generate IStrings
- Accepts an optional parameter, a reference to an IStringGeneratorFn on its constructor

Use the subclasses IStringGeneratorMemberFn and IStringGeneratorRefMemberFn or create your own subclass to represent functions. These subclasses do two things:

- On the constructor, save a pointer to the function the object represents
- Implement stringFor to call the stored function

These objects are reference-counted to manage their destruction.

IStringGeneratorMemberFn and IStringGeneratorRefMemberFn objects represent member functions of the template class type. Review these classes for example usage.

Class header file: istrgen.hpp

IStringGeneratorFn - Hierarchy List

IMRefCounted

IStringGeneratorFn

IStringGeneratorMemberFn

IStringGeneratorasString

IStringGeneratorRefMemberFn

IStringGeneratorFn - Member Functions and Data by Group

Constructors & Destructor

You do not create objects of this class. It is an abstract base class.

~IStringGeneratorFn

```
public:virtual ~IStringGeneratorFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStringGeneratorFn

```
public:IStringGeneratorFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IStringGeneratorFn object.

Generate String

Establishes the protocol for generating an IString for the object parameter. Subclasses implement this function to generate an IString for that object.

stringFor

```
public:virtual IString stringFor(const T& object) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Is a pure virtual function. It accepts an object reference as a parameter; it is this object for which subclass objects generate a string. IStringGeneratorFn subclasses override this function to call the appropriate member function with the appropriate number of parameters.

IStringGeneratorFn - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
`addRef`	57	`count`	58
`removeRef`	57		

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

IStringGeneratorMemberFn

The IStringGeneratorMemberFn template class is an IStringGeneratorFn-derived class. It dispatches C++ member functions against an object to generate an IString. These objects represent member functions of class **T**, where **T** is the template argument. Objects of this class apply the stored member function against class **T** objects in the stringFor member function, where object is a reference to a class **T** object.

The constructor for these objects requires a pointer to a class **T** member function; this member function returns an IString and accepts no arguments. Subclass this class to support member functions with additional parameters.

The following example creates an IStringGeneratorMemberFn object:

```
class MyClass {
public:
  IString myString()
  {
   // Code to generate a string
  }
 // ......
};
//...
MyClass myObj;
IStringGeneratorMemberFn<MyClass> * genFunction =  new
   IStringGeneratorMemberFn<MyClass>( MyClass::myString );
```

Note: Objects of the IStringGeneratorMemeberFn class require the object passed on the stringFor member function not be a pointer. IStringGeneratorRefMemberFn objects generate strings from pointers to objects.

Class header file: istrgen.hpp

IStringGeneratorMemberFn - Hierarchy List

IMRefCounted

IStringGeneratorFn

IStringGeneratorMemberFn

IStringGeneratorMemberFn - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IStringGeneratorMemberFn

```
public:virtual ~IStringGeneratorMemberFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStringGeneratorMemberFn

```
public:IStringGeneratorMemberFn(IString ( T::* member ) ( ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create from a pointer to a member function of class **T**, where **T** is the template argument. The member function must return an IString and accept no arguments; also, it is not a const function.

member
A pointer to the member function.

Generate String

Use this member function to generate an IString from an object.

stringFor

```
public:virtual IString stringFor(const T& object) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the appropriate member function on the object reference parameter. The called member function is initialized from the parameter on the IStringGeneratorMemberFn constructor.

object Reference to the object against which it dispatches the member function.

Note: The object parameter must be an object of type **T** (where **T** is the template class argument) and not a pointer.

IStringGeneratorMemberFn - Inherited Member Functions and Data

Inherited Public Functions

IStringGeneratorFn			
Definition	**Page Number**	**Definition**	**Page Number**
~IStringGeneratorFn stringFor	538 538	IStringGeneratorFn	538

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

IStringGeneratorRefMemberFn

The IStringGeneratorRefMemberFn template class is an IStringGeneratorFn-derived class. It dispatches C++ member functions against an object pointer to generate an IString. These objects represent member functions of class U to be applied against objects of type **T**, where **T** and **U** are template arguments and **T** is defined as typedef U* T;. This makes objects of type **T** the pointers to objects of class **U**. Objects of this class apply the stored member function against type **T** objects in the stringFor(const T& object) function, where object is a reference to a type **T** object.

The constructor for these objects requires a pointer to a class **U** member function; this member function returns an IString and accepts no arguments. Derive a class from this class to support member functions with additional parameters.

The following example creates an IStringGeneratorRefMemberFn object:

```
class MyClass {
public:
  IString myString()
  {
   // Code to generate a string
  }
 // ......
};
//...
typedef  MyClass *  pMyClass;
MyClass myObj;
IStringGeneratorRefMemberFn<MyClass> * genFunction =  new
   IStringGeneratorRefMemberFn<MyClass>( pMyClass, MyClass::myString );
```

Note: Objects of the IStringGeneratorRefMemberFn class require that the object passed on the stringFor member function is a pointer or an object that supports the operator->. IStringGeneratorMemberFn objects generate strings from objects that are not pointers.

Class header file: istrgen.hpp

IStringGeneratorRefMemberFn - Hierarchy List

IMRefCounted

IStringGeneratorFn

IStringGeneratorRefMemberFn

IStringGeneratorRefMemberFn - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IStringGeneratorRefMemberFn

```
public:virtual ~IStringGeneratorRefMemberFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStringGeneratorRefMemberFn

```
public:IStringGeneratorRefMemberFn(IString ( U::* member ) ( ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create from a pointer to a member function of class **U**, where **U** is a template argument. The member function must return an IString, accept no arguments, and be a non-const function.

member
A pointer to the member function.

This constructor requires a pointer to a class U member function. This member function should return an IString, accept no parameters and be a non-const function.

Generate String

Use this member function to generate an IString from an object pointer.

stringFor

```
public:virtual IString stringFor(const T& object) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function calls the appropriate member function on the object reference parameter. The called member function is initialized from the parameter on the IStringGeneratorMemberFn constructor. The object parameter should be a pointer.

object Reference to the object against which it dispatches the member function.

Note: The object parameter must be an object of type **T**, which is a pointer to an object of type **U** (where **T** and **U** are the template class arguments) and class **U** contains the member function that is dispatched.

IStringGeneratorRefMemberFn - Inherited Member Functions and Data

Inherited Public Functions

IStringGeneratorFn			
Definition	**Page Number**	**Definition**	**Page Number**
~IStringGeneratorFn	538	IStringGeneratorFn	538
stringFor	538		

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef	57	count	58
removeRef	57		

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted	57	IMRefCounted	57
operator =	57		

Inherited Protected Data

None

IStringParser

The IStringParser class parses the content of an IString and places portions of the string into other strings. You can limit the parsing of a string by specifying the following:

- Patterns that must be matched
- Relative or absolute column numbers

This class's functions work much like the REXX parse statement.

Typically, you create IStringParser objects implicitly by applying the right-shift operator to an IString. IStringParser also provides the right-shift operator as a member function so you can chain together invocations of the operator. For example, a typical expression using IStringParser objects might look like the following:

```
IString aFileName("c:\test\dir");
IString drive, path;
aFileName >> drive >> ':' >> path;
```

After this, drive would be equal to ″c″ and path would be equal to ″\test\dir″.

The right-shift operator (>>) does one of four things, depending on the type of the right-hand operand:

IString
The string parser object sets this string to the next token from the text being parsed.

pattern
The parser advances to the next character beyond the occurrence of that pattern in its text. The pattern can be any of the following:

const char*
Searches for the sequence of characters described by the character array.

const IString
Searches for the sequence of characters described by the string. Note that the treatment of a const IString is fundamentally different from the treatment of a non-const IString.

char Searches for the next occurrence of the specified character.

IStringTest
Searches for the next character in the text for which the string test object returns true.

number
The current parser text position is adjusted by the specified amount. The value can be positive or negative. For example:

```
"1234" >> token1 >> 1 >> token2 >> 2 >> token3;
```

This results in the following:

```
token1 == "1"
token2 == "23"
token3 == "4"
```

special
IStringParser defines special right-shift operands that perform the following special-purpose parser operations:

IStringParser::reset
This enumerator resets the parser text position to 1.

IStringParser::skip
This enumerator skips one token in the text. It is equivalent to **>> temp**, where temp is a temporary IString that is discarded. This is equivalent to using '.' in REXX.

IStringParser::Skip
An object of this class skips a given number of tokens.

You can also use the left-shift operator with an unsigned numeric parameter. This repositions the parser object to the specified column. Note that the parameter is not relative as it is in the case of the right-shift operator. Instead, it is an absolute column position.

Class header file: istparse.hpp

IStringParser - Hierarchy List

IStringParser

IStringParser - Member Functions and Data by Group

Constructors & Destructor

You can construct a string parser object by providing the following:

- A string that defines the text to be parsed
- An existing parser object (copy constructor)

You construct parser objects by applying the right-shift operator to a string. The constructor is protected to prevent you from creating objects except via use of those operators. Creation is prevented because of the nature of string parser objects. Because they hold references to operands, it is unwise to permit the objects to persist beyond the scope of those operands.

~IStringParser

```
public:~IStringParser()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys a string parser object.

IStringParser

Constructs a string parser object.

Overload 1

```
protected:IStringParser(const IStringParser& parser)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an object from an existing IStringParser object. The IStringParser object specifies the text string to parse. This constructor increments the usage count of the IStringParser object.

Overload 2

```
protected:IStringParser(const IString& text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an object from an IString object. The IString object specifies the text string to parse.

Absolute Column Positioning

Use this member function to reset the parser text position to an absolute column number.

operator <<

```
public:IStringParser& operator <<(ioc::unumeric_t position)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes the parser text position to an absolute column number. This is a left-shift operator.

Parsing

These member functions provide string parsing services.

operator >>

Parses the text string. The right-shift operator is the primary function for parsing the text string. The Open Class Library overloads this function, so you can specify how you want the text string parsed via the type of parameter accepted by a particular overload.

Overload 1

```
public:IStringParser& operator >>(const SkipWords& skipObject)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Skips the next **n** words in the parser text, where **n** is the number of words specified when constructing the IStringParser::SkipWords object.

Overload 2

```
public:IStringParser& operator >>(Command command)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the parser text position as follows:
- To the beginning of the text
- To skip the next token in the parser text

Use the enumeration IStringParser::Command to specify the parsing token.

Pattern Matching

Use these member functions to advance to the next occurrence of the argument pattern in the parser text. Upon return, the parser is positioned at the next character beyond the text that matched the pattern. If the pattern is not found, the parser is positioned off the end of the text.

When using an IString as a pattern, you should cast it to a const IString reference.
operator >>

Parses the text string. The right-shift operator is the primary function for parsing the text string. The Open Class Library overloads this function, so you can specify how you want the text string parsed via the type of parameter accepted by a particular overload.

Overload 1

```
public:IStringParser& operator >>(const IStringTest& test)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Applies the IStringTest object to the parser text and moves the parser text position to the next character that satisfies the string test. If the string test is not satisfied, the parser moves the position off the end of the parser text.

Overload 2

```
public:IStringParser& operator >>(const char* pattern)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds a matching pattern within the parser text and moves the parser text position. If the pattern is not found, the parser moves the position off the end of the parser text.

Overload 3

```
public:IStringParser& operator >>(const IString& pattern)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds a matching pattern within the parser text and moves the parser text position. If the pattern is not found, the parser moves the position off the end of the parser text.

Overload 4

```
public:IStringParser& operator >>(char pattern)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds a matching pattern within the parser text and moves the parser text position. If the pattern is not found, the parser moves the position off the end of the parser text.

Relative Column Positioning

Use these members to move the parser text position relative to its current position. A negative argument moves backward; a positive argument moves forward. The adjustment is made starting at the point at which the prior parsing instruction started.

For example:

```
"1234" >> token1 >> 1 >> token2 >> 2 >> token3;
```

This results in the following:

```
token1 == "1"
token2 == "23"
token3 == "4"
```

operator >>

Parses the text string. The right-shift operator is the primary function for parsing the text string. The Open Class Library overloads this function, so you can specify how you want the text string parsed via the type of parameter accepted by a particular overload.

Overload 1

```
public:IStringParser& operator >>(ioc::unumeric_t delta)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IStringParser& operator >>(int delta)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the parser text position relative to the current parser text position.

Tokens

Use these members to parse the next token from the parser object and place it into the IString operand. By necessity, these members place the rest of the parser text into the string. When the parser encounters a subsequent parsing instruction, it goes back and adjusts the token placed into the string.

For example:

```
 "token1 token2" >> token1    // token1 == "token1 token2" at this point
                 >> token2;   // token2 == "token2" and
                             // token1 == "token1"
```

operator >>

```
public:IStringParser& operator >>(IString& token)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Parses the text string. The right-shift operator is the primary function for parsing the text string. The Open Class Library overloads this function, so you can specify how you want the text string parsed via the type of parameter accepted by a particular overload.

Parses the next token from the object into the IString object. This parameter places the rest of the parser text into the IString object. When the parser encounters a subsequent parsing instruction, it adjusts the token placed into the string.

IStringParser - Enumerations

Command

These enumerators specify special-purpose parsing tokens:

reset Resets the parser position to 1.

skipWord
Causes the parser to skip one token (that is, a word) in the input text.

IStringParser - Associated Globals

operator <<

```
IStringParser
  operator <<( const IString& text,
               ioc::unumeric_t position )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IStringTest& test )
```

Implicitly constructs an IStringParser object from a string and performs pattern matching based on the IStringTest object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IStringParser::SkipWords& skipObject )
```

Implicitly constructs an IStringParser object from a string and skips the number of words specified by the IStringParser::SkipWords object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const char* pattern )
```

Implicitly constructs an IStringParser object from a string and performs pattern matching on the object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               IStringParser::Command command )
```

Implicitly constructs an IStringParser object from a string and applies the IStringParser::Command to the object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               int position )
```

Implicitly constructs an IStringParser object from a string and performs relative positioning on the object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               ioc::unumeric_t position )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               IString& token )
```

Implicitly constructs an IStringParser object from a string and parses the text into the specified token string object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IString& pattern )
```

Implicitly constructs an IStringParser object from a string and performs pattern matching on the object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               char pattern )
```

Implicitly constructs an IStringParser object from a string and performs pattern matching on the object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStringParser - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStringParser::SkipWords

The nested class IStringParser::SkipWords skips a specified number of words in the input text without assigning those words to output strings. Use these objects when parsing text with the class IStringParser.

Class header file: istparse.hpp

IStringParser::SkipWords - Hierarchy List

IStringParser::SkipWords

IStringParser::SkipWords - Member Functions and Data by Group

Constructors & Destructor

You can construct objects of this class by specifying the number of words to skip. Use in conjunction with IStringParser objects to parse the content of an IString and place portions of the string into other strings.

SkipWords

```
public:SkipWords(ioc::unumeric_t numberOfWords = 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You can construct objects of this class by specifying the number of words to skip. The default is one word.

Word Functions

Use this member function to retrieve the number of words to skip. You set the number of words to skip in the constructor.

numberOfWords

```
public:ioc::unumeric_t numberOfWords() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of words to skip.

IStringParser::SkipWords - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStringTest

The IStringTest class defines the basic protocol for test objects that you can pass to IString functions or I0String functions to assist in performing various test and search functions. This class also provides concrete implementation for the common case of using a C/C++ function for such testing.

The Open Class Library provides a derived template class, IStringTestMemberFn, to facilitate using member functions of any class on the IString functions that support IStringTest.

Derived classes should reimplement the virtual function IStringTest::test to test characters passed by the IString and return the appropriate result.

A constructor for this class accepts a pointer to a C/C++ function that in turn accepts an integer as a parameter and returns a boolean. You can use such functions anywhere an IStringTest can be used. Note that this is the type of the standard C Library ″is″ functions that check the type of C characters.

Windows Considerations

If I18N (Internationalization) semantic is turned on, the integer parameter passed to the test function will be the wide-character representation of the character being tested.

Class header file: istrtest.hpp

IStringTest - Hierarchy List

IStringTest

IStringTestMemberFn

IStringTest - Member Functions and Data by Group

Constructors & Destructor

You can construct and destruct objects of this class with a pointer to the C/C++ function to be used to implement the member IStringTest::test. Such members can be used anywhere an IStringTest can be used. Note that these members are the same as the standard C library ″is″ functions that check the type of C characters.

This class also provides a protected constructor, which derived classes can use to reuse the space for the C/C++ function pointer.

~IStringTest

```
public:virtual ~IStringTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the string test object.

IStringTest

Constructs a string test object.

Overload 1

```
public:IStringTest(CFunction cFunc)
```

Overload 2

```
public:IStringTest(_mscwcharfunc cFunc)
```

Overload 3

```
public:IStringTest(CPPFunction& cppFunc)
```

Overload 4

```
public:IStringTest(CFunction& cFunc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Accepts a pointer to a C function.

Overload 5

```
protected:IStringTest(FnType type, void* userData)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Accepts a function type and a pointer to user data.

Function Types

Use these typedefs to identify the test function.

type

```
protected:FnType type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member FnType. FnType is an enumeration describing the various flavors of functions supported: user-defined, C, C++ static or nonmember function, C++ member function, or const C++ member function.

Testing

Use this member function to implement an actual test.

test

```
public:virtual bool test(int c) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests the specified integer (character) and returns true or false as returned by the C/C++ function provided at construction. Derived classes should override this function to implement their own testing function.

IStringTest - Enumerations

FnType

Use these enumerators to specify the type of functions supported:

user User-defined.

c C.

cpp C++ static or non-member function.

memFn
C++ member function.

cMemFn
Const C++ member function.

IStringTest - Type Definitions

(int)

```
typedef bool CPPFunction ( int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

(unsigned short)

```
typedef bool _mscwcharfunc ( unsigned short )
```

CFunction

```
typedef ICStrTestFn CFunction
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Pointer to the C/C++ function that accepts an integer parameter and returns a boolean.

IStringTest - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStringTestMemberFn

The IStringTestMemberFn class is a template class that provides an IStringTest-type wrapper for particular C++ member functions. You can use such member functions in conjunction with functions from IString and I0String that accept an IStringTest object as an parameter.

Customization (Template Argument)

IStringTestMemberFn is a template class that is instantiated with the following template argument:

T The class of object whose member function is to be wrappered.

Class header file: istrtest.hpp

IStringTestMemberFn - Hierarchy List

IStringTest

IStringTestMemberFn

IStringTestMemberFn - Member Functions and Data by Group

Constructors & Destructor

You can construct objects of this class in the following ways:

- Use the constructor that supports const member functions.
- Use the constructor that supports non-const member functions. You must specify a non-const member function as the first parameter.

Both constructors for the object require the following:

- An object of the class T (non-const object for non-const member functions).
- A pointer to a member function of the class T. The Open Class Library applies this member function to the specified object to test each character passed to the test member of this class. The member function must accept a single integer parameter and return a boolean.

IStringTestMemberFn

Constructs an IStringTestMemberFn object.

Overload 1

```
public:IStringTestMemberFn(T& object, NonconstFn nonconstFn)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this for the non-const member functions. The object of the class T must be non-const.

Overload 2

```
public:IStringTestMemberFn(const T& object, ConstFn constFn)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this for the const member functions.

Testing

Use these member functions to dispatch member functions.

test

```
public:virtual bool test(int c) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overridden to dispatch a member function against an object.

IStringTestMemberFn - Type Definitions

ConstFn

```
typedef bool ( T::* ConstFn)( int )  const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A prototype for the test function, which works on const strings.

NonconstFn

```
typedef bool ( T::* NonconstFn)( int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A prototype for the test function, which works on non-const strings.

IStringTestMemberFn - Inherited Member Functions and Data

Inherited Public Functions

IStringTest			
Definition	**Page Number**	**Definition**	**Page Number**
~IStringTest	551	IStringTest	551

Inherited Public Data

None

Inherited Protected Functions

IStringTest			
Definition	**Page Number**	**Definition**	**Page Number**
IStringTest	551		

Inherited Protected Data

IStringTest			
Definition	**Page Number**	**Definition**	**Page Number**
type	552		

ITabRulerStyle

The ITabRulerStyle class is a style that controls the tab stops of the paragraphs it is applied to.

Class header file: istyles.hpp

ITabRulerStyle - Hierarchy List

ITextStyle

IParagraphStyle

ITabRulerStyle

ITabRulerStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITabRulerStyle objects.

~ITabRulerStyle

```
public:virtual ~ITabRulerStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITabRulerStyle object.

ITabRulerStyle

Constructs an ITabRulerStyle object.

Overload 1

```
public:ITabRulerStyle(const ITabRulerStyle& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITabRulerStyle(GCoord autoSpacing)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a tab ruler style that contains only autotabs at the specified interval.

Overload 3

```
public:ITabRulerStyle( ioc::isize_t tabCount,
                       const ITabStop* tabs,
                       GCoord autoSpacing )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a tab ruler style according to the following parameters:

tabCount
Number of tab stops in array pointed to by tabs.

tabs Pointer to an array of ITabStop. tabCount tab stops are copied from this array into the new style.

autoSpacing
Spacing interval for autotabs.

Overload 4

```
public:ITabRulerStyle(const IMTabRuler& ruler)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
protected:ITabRulerStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Getting and Setting the Tab Ruler

Use these member functions to set the tab stops, replicating those of the given ruler, and to get the current tab ruler settings.

ruler

```
public:IMTabRuler* ruler()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current tab ruler.

setRuler

```
public:void setRuler(const IMTabRuler& ruler)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The tab stops in the ruler are copied, and any tabs stops originally in the style are lost.

ITabRulerStyle - Inherited Member Functions and Data

Inherited Public Functions

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~IParagraphStyle	469	propagates	469

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITabRulerStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	555	getValueFromSet	555
isContainedInSet	555	isValueContainedInSet	555
removeFromSet	555		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

IParagraphStyle			
Definition	**Page Number**	**Definition**	**Page Number**
IParagraphStyle	469	operator =	469

Inherited Protected Data

None

ITabStop

ITabStop represents a single tab stop in a tab ruler. Typically, you create an ITabStop with a specified co-ordinate and store them in an IStandardTabRuler object. The IStandardTabRuler class controls the relative positions of different tab stops.

Class header file: imtabrl.hpp

ITabStop - Hierarchy List

ITabStop

ITabStop - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITabStop objects.

~ITabStop

```
public:virtual ~ITabStop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITabStop object.

ITabStop

Constructs an ITabStop object.

Overload 1

```
public:ITabStop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor. Creates a tab stop at position 0 of type kLeft.

Overload 2

```
public:ITabStop(GCoord position, ETabStopKind type)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a tab stop object based on the passed-in parameters.

Overload 3

```
public:ITabStop(const ITabStop& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor. Use this constructor to replaces this, the current, ITabStop object with the given one.

Assignment Operator

Use this operator to replace the current ITabStop object with the given one.

operator =

```
public:ITabStop& operator =(const ITabStop& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Comparing ITabStop Objects

Use these member functions to test two ITabStop objects to determine if their contents are the same or different.

operator !=

```
public:bool operator !=(const ITabStop& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITabStop object against the current one to determine if their contents are different.

operator ==

```
public:bool operator ==(const ITabStop& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITabStop object against the current one to determine if their contents are equal.

Obtaining a Tab Stop's Type or Position

Use these member functions to get the type of a tab stop or its position in global coordinates.

position

```
public:GCoord position() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the position of the tab stop in its global coordinates.

type

```
public:ETabStopKind type() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the type of tab stop. The type is identified by one of the constants defined by the ETabStopKind enumerator.

Streaming ITabStop Objects

Use these member functions to read the ITabStop object in from a specified stream or write it out to one.

readFromStream

```
public:virtual void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads the ITabStop object in from the specified data stream.

writeToStream

```
public:virtual void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the ITabStop object out to the specified data stream.

ITabStop - Enumerations

ETabStopKind

Identifies the type of tab stop:

- kLeft - Text is left-justified at the tab stop position.
- kCenter - Text is centered at the tab stop position.
- kRight - Text is right-justified at the tab stop position.
- kDecimal - The first decimal-point character in the text is aligned to the tab stop position.

- kAuto - An automatically-generated tab stop, instead of one set by the client. Works the same as kLeft.

ITabStop - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IText

The IText class provides a variable length, styled Unicode text storage object.

This class manages the storage automatically, and provides a protocol to access and change the characters and styles. The protocol is a superset of the ANSI standard basic_string protocol; the protocol always represents the characters using Unicode.

Generally, ITexts are created on the stack rather than the heap. An IText is merely a smart pointer to a separate internal storage object which is allocated on the heap, so it's generally not necessary to allocate the IText there as well. Also, because IText is just a smart pointer, copy and assignment performance are guaranteed to be good enough to allow IText to be used with value semantics when used as a function parameter or return value.

If an IText object constructed with an IString or a null-terminated character array, it initializes its internal Unicode data only when Unicode is needed. This allows fast construction and copying of IText objects constructed with 8-bit character strings. For this type of costruction, the IText object can be associated with a character code set. This character code set will be used to convert the 8-bit data to Unicode. If a character code set is not provided, IText will use the code set associated with the current locale. If there is any problem during the conversion, IText assumes the ISO8859-1 code set and redoes the conversion.

Do not derive from this class.

Subclassing

Class header file: itext.hpp

IText - Hierarchy List

IText

IText - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IText objects.

~IText

```
public: ~IText()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an IText object.

IText

Constructs an IText object.

Overload 1

```
public:IText(const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs and initializes an IText from another IText (copy constructor).

Because an IText is merely a pointer to a separate storage object, which can be shared among ITexts, copy constructor performance is guaranteed to be fast enough to allow IText to be used with value semantics when used as a function parameter or return value.

The character code set information is inherited from that.

that The IText to copy (both characters and styles are copied).

Overload 2

```
public:IText( const IText& that,
       offset_type thatPos,
       length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs and initializes an IText from a substring of another IText. The character code set information is inherited from that.

that The IText containing the text to copy (both characters and styles are copied).

thatPos
The position of the first character in ″that″ to copy into the new IText.

thatLength
The number of characters in the subrange of ″that″ to copy into the new IText, or npos, which specifies all characters in ″that″ from thatPos to the end.

Exception

Exception	Description
IInvalidParameter	Thrown if thatPos > that.length()

Overload 3

```
public:IText(const IString& that, const char* codeset = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs and initializes an IText from an IString.

If codeset is 0 (the default), the IText object does not assume a character code set for the incoming IString. When transcoding is

required (typically when Unicode data is needed), the IText object uses the character code set of the current locale to transcode the string.

If codeset is supplied, IText stores the character code set and uses that to transcode the string.

that The string of characters used to initialize the text object.

codeset
The encoding scheme used to encode the string (e.g. ISO8859-1, IBM-932, UTF-8, etc.). The portable character code sets (such as Shift-JIS) are also supported.

Overload 4

```
public:IText(const ioc::unichar_t* that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IText( const ioc::unichar_t that [ ],
       length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IText(ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:IText(length_type thatCopies, ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IText(const IString& that, const ICCSId& id)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:IText(const char* that, const ICCSId& id)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 10

```
public:IText(const char* that, const char* codeset = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs and initializes an IText from a null-terminated array of char.

If codeset is 0 (the default), the IText object does not assume a character code set for the incoming char data. When transcoding is required (typically when Unicode data is needed), the IText object uses the character code set of the current locale to transcode the string.

If codeset is supplied, IText stores the character code set and uses that to transcode the string.

that The null-terminated array of chars used to initialize the object.

codeset
The encoding scheme used to encode the string (e.g. ISO8859-1, IBM-932, UTF-8, etc.). The portable character code sets (such as Shift-JIS) are also supported.

Overload 11

```
public:IText()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an empty IText.

Overload 12

```
public:IText(ITextIterator thatFrom, ITextIterator thatTo)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs and initializes an IText from a substring of another IText by using the being and end ITextIterators. The character code set information is inherited from the IText object associated with the iterator.

thatFrom
The first character to copy.

thatTo The character after the last character to copy.

Exception

Exception	Description
IInvalidParameter	Thrown if thatFrom or thatTo don't form a proper range

Accessing Characters

Use these member functions to obtain a reference to a character or replace a character in the current IText object.

at

Same as operator[](pos).

Overload 1

```
public:ioc::unichar_t at(offset_type pos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pos The given position.

Overload 2

```
public:ICharacterReference at(offset_type pos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pos The given position.

at_put

```
public:void at_put(offset_type pos, ioc::unichar_t newChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the character at the specified position with the specified character. The new character does not take on the styles of the characters around it, or the styles of the character it replaces.

pos The position of the character the be replaced.

newChar

The replacement character.

operator []

Returns a reference to the character at the specified position in the IText.

Overload 1

```
public:ioc::unichar_t operator [](offset_type pos) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character at the specified position in the IText.

pos The given position.

Return

If pos == length(), UUnicodeSpecial::kNull; otherwise, the specified character.

Exception

Exception	Description
IInvalidParameter	Thrown if pos > length()

Overload 2

```
public:ICharacterReference operator [](offset_type pos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pos The given position.

Return

An ICharacterReference referring to the specified character. The ICharacterReference returned by this function is only guaranteed to refer to the correct character until the next call to c_str(), data(), or a non-const member of this IText. (ICharacterReference is semantically equivalent to a live ioc::unichar_t&, but safer.)

Adding and Removing Styles

Use these member functions to add and remove styles from the IText object.

addStyles

Applies the specified style to the specified range. The specified styles supercede any of the same class already applied anywhere within the specified range. If the specified styles include paragraph styles, they are applied to the whole paragraphs containing the specified range.

Overload 1

```
public:void
  addStyles( const ITextStyle& style,
             offset_type starting = 0,
             length_type count = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version of the function applies a single style to the specified range.

style The style to apply.

starting
The offset of the first character in the range to apply the style to.

count The number of characters in the range to apply the style to. npos may be used for this parameter, and means ″add these styles to all characters from starting to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

Overload 2

```
public:void
  addStyles( const ITextStyleSet& styles,
             offset_type starting = 0,
             length_type count = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version of the function allows the user to apply multiple styles in a single step. It is equivalent to calling addStyles() repeatedly, once for each style in the style set.

style A style set containing the styles to apply.

starting
The offset of the first character in the range to apply the styles to.

count The number of characters in the range to apply the styles to. npos may be used for this parameter, and means ″add these styles to all characters from starting to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

removeStyles

Removes any styles matching the classes of the specified style or styles from the specified range. Only the classes of the specified styles are significant; their values are ignored. If the specfied styles include paragraph styles, the matching paragraph styles are removed from the whole paragraphs containing the specified range.

Overload 1

```
public:void
  removeStyles( offset_type starting = 0,
                length_type count = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version removes ALL styles applied to the specified range of text (which means it removes all paragraph styles from the paragraphs containing the specified range).

starting
The offset of the first character in the range to remove the styles from.

count The number of characters in the range to remove the styles

from. npos may be used for this parameter, and means ″remove these styles from all characters from starting to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

Overload 2

```
public:void
  removeStyles( const ITextStyleSet& styles,
                offset_type starting = 0,
                length_type count = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version allows the user to remove several styles in a single step. It is equivalent to calling removeStyles() repeatedly, once for each style in the style set.

style A style set containing the styles to remove (all styles matching the class of a style in this set are removed; it doesn't matter if their values also match).

starting
The offset of the first character in the range to remove the styles from.

count The number of characters in the range to remove the styles from.

count npos may be used for this parameter, and means ″remove these styles from all characters from starting to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

Overload 3

```
public:void
  removeStyles( const ITextStyle& style,
                offset_type starting = 0,
                length_type count = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

style The style to remove (all styles matching this style's class are removed; it doesn't matter if their values match this style's value).

starting
The offset of the first character in the range to remove the style from.

count The number of characters in the range to remove the style from. npos may be used for this parameter, and means ″remove these styles from all characters from starting to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

Appending Characters

Use these member functions to append characters to the IText object.

append

Appends characters at the end of the IText. The new characters retain their character styles (raw Unicode data are considered unstyled), and paragraph styles are propagated. It is legal to append characters from an IText into itself.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText& append(const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

that The text to append (both styles and characters are appended). May be the same as *this.

Exception

Exception	Description
IInvalidRequest	Thrown if the resulting length would be greater than max_size().

Overload 2

```
public:IText& append(const ioc::unichar_t* that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IText& append(ITextIterator thatFrom, ITextIterator thatTo)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

thatFrom
An iterator referring to the first character in the source string (both characters and styles are appended).

thatTo An iterator referring to the position AFTER the last character in source string.

Exception

Exception	Description
IInvalidParameter	Thrown if thatFrom and thatTo do not form a proper range.

Overload 4

```
public:IText&
  append( const ioc::unichar_t that [ ],
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IText& append(length_type thatCopies, ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IText&
  append( const IText& that,
          offset_type thatPos,
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

that The IText containing the text to append (both styles and characters are appended). May be the same as *this.

thatPos
The position of the first character in ″that″ to append.

thatLength
The number of characters in ″that″ which we want to append, or npos, which specifies all characters in ″that″ from thatPos to the end.

Exception

Exception	Description
IInvalidParameter	Thrown if thatPos > that.length().

operator +=

Appends characters at the end of the IText. The new characters retain their character styles (raw UCS2Char data are considered unstyled), and paragraph styles are propagated. It is legal to append characters from an IText into itself.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText& operator +=(const ioc::unichar_t* that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText& operator +=(ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IText& operator +=(const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operators and Functions

Use these operators and member functions to replace the IText contents--its characters and styles--with the given values.

assign

Replaces the entire contents of *this (characters and styles) with the specified text.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText&
  assign( const IText& that,
          offset_type thatPos,
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

that An IText containing the new text. May be the same as *this.

thatPos
The position of the first character in ″that″ to copy into *this.

thatLength
The number of characters from ″that″ to copy into *this, or npos, which specifies all characters in ″that″ from thatPos to the end.

Exception

Exception	Description
IInvalidParameter	Thrown if thatPos > that.length().

Overload 2

```
public:IText& assign(length_type thatCopies, ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IText& assign(const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

that The new text. May be the same as *this (which would do nothing).

Overload 4

```
public:IText& assign(const ioc::unichar_t* that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IText&
  assign( const ioc::unichar_t that [ ],
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IText& assign(ITextIterator thatFrom, ITextIterator thatTo)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

thatFrom
An iterator referring to the first character in the source string.

thatTo An iterator referring to th position AFTER the last character in the source string.

Exception

Exception	Description
IInvalidParameter	Thrown if thatFrom and thatTo do not form a proper range.

operator =

Replaces the entire contents of *this (characters and styles) with the specified text.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText& operator =(const ioc::unichar_t* that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText& operator =(ioc::unichar_t that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IText& operator =(const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Comparing the IText Object with Other Entities

Use these member functions to compare the current IText object with the given entities and values.

compare

This function compares this IText to another IText. This is a bitwise lexicographic compare: The result of the comparison is the result of comparing the numeric (Unicode) values of the first pair of nonmatching characters between the strings. If one string is a proper substring of the other, the shorter string compares less. Styles are not taken into account by the compare() functions.

Overload 1

```
public:int compare(const ioc::unichar_t* that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:int
  compare( offset_type thisPos,
           length_type thisLength,
           const IText& that,
           offset_type thatPos,
           length_type thatLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares a subrange of this IText to a subrange of another IText.

thisPos
The offset of the first character from *this to use in the comparison.

thisLength
The length of the subrange of *this to use in the comparison, or npos, which specifies all characters in *this from thisPos to the end.

that The IText containing the characters to compare against. May be the same as *this.

thatPos
The offset of the first character from ″that″ to use in the comparison.

thatLength
The length of the subrange of ″that″ to use in the comparison, or npos, which specifies all characters in ″that″ from thatPos to the end.

Return
0 if the two strings are equal, -1 if the specified subrange of *this is less than the specified subrange of ″that,″ and 1 if the specified subrange of *this is greater than the specified subrange of ″that.″

Exception

Exception	Description
IInvalidParameter	Thrown if thisPos > this->length() or thatPos > that.length().

Overload 3

```
public:int
  compare( offset_type thisPos,
           length_type thisLength,
           const ioc::unichar_t that [ ],
           length_type thatLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:int
  compare( offset_type thisPos,
           length_type thisLength,
           const IText& that ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares a subrange of this IText to another IText.

thisPos
The offset of the first character from *this to use in the comparison.

thisLength
The length of the subrange of *this to use in the comparison, or npos, which specifies all characters in *this from thisPos to the end.

Return
0 if the two strings are equal, -1 if the specified subrange of *this is less than ″that,″ and 1 if the specified subrange of *this is greater than ″that.″

Exception

Exception	Description
IInvalidParameter	Thrown if thisPos > this->length().

Overload 5

```
public:int compare(const IText& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares this IText to another IText.
that The IText to compare against.
Return
0 if the two strings are equal, -1 if *this is less than ″that,″ and 1 if *this is greater than ″that.″

hash

```
public:ioc::numeric_t hash() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Generates a hash value for the characters in the IText. Styles are not significant when computing a hash value. The hash value is cached for performance.

Converting Text

Use these member functions to return converted text.

operator const char *

```
public:operator const char *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to an array of char that is equivalent to the contents of ″this″ transcoded into the current code page. To convert the text to any other encoding standard, use a transcoder. The storage is owned by the IText, and the pointer is guaranteed to be good only until the next call to c_str(), data(), or a non-const member of the IText.

operator const ioc::unichar_t *

```
public:operator const ioc::unichar_t *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as c_str().

Debugging

Use these member functions to debug the IText object.

printDebugInfo

```
public:void printDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Dumps the characters (transcoded to ISO 8859-1) and styles, potentially along with some other internal values, in readable form suitable for debugging.

Deleting Characters

Use these member functions to delete characters from the IText object.

erase

The characters following the deleted ones are moved up to ″fill up the hole,″ and styles are re-propagated.

Overload 1

```
public:IText&
  erase( offset_type pos = 0,
         length_type numChars = npos )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes characters from an IText.

pos The offset of the first character to delete.

numChars

The number of characters to delete, or npos, which causes all characters from ″pos″ to the end to be deleted.

Exception

Exception	Description
IInvalidParameter	Thrown if pos > length().

Overload 2

```
public:ITextIterator erase(ITextIterator pos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes a single character from an IText.

pos An iterator referring to the character to delete.

Return

pos (which is not changed, and now refers to the first character after the deleted one)

Exception

Exception	Description
IInvalidParameter	Thrown if pos doesn't refer to a valid character in *this.

Overload 3

```
public:ITextIterator erase(ITextIterator from, ITextIterator to)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes characters from an IText.

from An iterator referring to the first character to delete.

to An iterator referring to the first character AFTER the range to delete (This character is not deleted, and will be at the position referred to by ″from″.)

Return

from (which is not changed, and now refers to the first character after the deleted ones)

Exception

Exception	Description
IInvaidParameter	Thrown if from and to do not form a proper range on *this..

Extracting a Substring

Use this member function to obtain an IText object containing the substring extracted from the current IText object.

substr

```
public:IText
  substr( offset_type pos = 0,
          length_type length = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a new IText whose contents are equal to the specified subrange of the original IText. (a.substr(x, y) is the same as IText(a, x, y))

pos Position of first character to extract.

length Number of characters to extract. ″npos″ can be used to mean ″all characters from pos to the end″

Exception

Exception	Description
IInvalidParameter	Thrown if pos >= length().

Extracting and Swaping Text

Use these member functions extract a range of text from an IText object as an array of Unicode characters (ioc::unichar_t) or to exchange the contents of two IText objects.

copy

```
public:length_type
  copy( ioc::unichar_t that [ ],
        length_type maxChars,
        offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a subrange of an IText into an array of Unicode characters (ioc::unichar_t).

that A pointer to an array of Unicode characters (ioc::unichar_t) that will receive the extracted characters

maxChars

The maximum number of characters to copy. This will generaly be either the length of the substring the caller is interested in, or the number of characters the caller has allocated storage for.

swap

```
public:void swap(IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Exchanges the contents of two ITexts. Character code set information is also swapped.

Getting Iterators

Use these member functions to obtain iterators that refer to various positions within the IText object.

begin

Returns an iterator that refers to the first character in the IText.

Overload 1

```
public:IConstTextIterator begin() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITextIterator begin()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

end

Returns the past-the-end iterator for the IText (i.e., an iterator that refers to the position immediately beyond the last character).

Overload 1

```
public:ITextIterator end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IConstTextIterator end() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

rbegin

Returns a reverse iterator referring to the last character in the IText.

Overload 1

```
public:IReverseConstTextIterator rbegin() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IReverseTextIterator rbegin()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

rend

Returns the past-the-beginning reverse iterator for the IText.

Overload 1

```
public:IReverseTextIterator rend()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IReverseConstTextIterator rend() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Getting Style Information

Use these member functions to obtain current style and style run information for positions or ranges within the ITextObject.

continuousStylesOver

Overload 1

```
public:bool
  continuousStylesOver( offset_type starting,
                        length_type count,
                        ITextStyleSet& styles ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns all the styles which are present and have the same value across the entire specified range.

starting
Starting offset of the range of interest.

count Length of the range of interest.

styles Receives copies of all styles that are present and have a consistent value across the entire range specified by ″starting″ and ″count.″ This style set is emptied out before being filled in with these styles.

Return
True if there are any styles that exist and have a consistent value across the entire range of interest, false if there aren't.

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

Overload 2

```
public:bool
  continuousStylesOver( offset_type starting,
                        length_type count,
                        ITextStyleSet& styles,
                        ITextStyle::EStylePropagation propagationToMatch ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns all the styles of the specified propagation type which are consistent across the entire specified range.

starting
Starting offset of the range of interest.

count Length of the range of interest.

styles Receives copies of all styles of the specified propagation type that are present and have a consistent value across the entire range specified by ″starting″ and ″count.″ This style set is emptied out before being filled in with these styles.

propagationToMatch
Propagation type of interest. ITextStyle::kMaximumPropagation matches styles of all propagation types.

Return
True if there are any styles of the specified propagation type that exist and have a consistent value across the entire range of interest, false if there aren't.

Exception

Exception	Description
IInvalidParameter	Thrown if starting >= length().

isStyled

```
public:bool isStyled() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if there are any styles applied to any part of the IText.

maximumStyleSpan

Returns the maximum range over which the specified style has the specified value.

Overload 1

```
public:bool
  maximumStyleSpan( offset_type position,
                    const ITextStyleSet& set,
                    offset_type& starting,
                    length_type& count ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum range over which the specified styles have the specified values.

Checks the styles at ″position.″ If they include all of the styles in ″set,″ and all of their values match the values of the style in ″set,″ returns true, and ″starting″ and ″count″ represent the range over which those styles have those values (even if other styles vary). If the styles at ″position″ don't include all of the styles in ″set,″ or if any corresponding styles don't have the same value, returns false, and ″starting″ and ″count″ represent the range over which the styles in the text don't match the styles in ″set.″

position
The position at which to check styles.

set The style set containing the sought styles.

starting
If the return value is true, receives starting offset of the range over which the specified styles have the specified values. If the return value is false, receives starting offset of the range over which the specified styles do not have the specified values.

count If the return value is true, receives length of the range over which the specified styles have the specified values. If the return value is false, receives the length of the range over which the specified styles do not have the specified values.

Return
True if all of the styles in ″set″ match styles in effect at ″position″ in both class and value, false otherwise.

Exception

Exception	Description
IInvalidParameter	Thrown if position >= length().

Overload 2

```
public:bool
  maximumStyleSpan( offset_type position,
                    const ITextStyle& style,
                    offset_type& starting,
                    length_type& count ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks the styles at ″position.″ If they include the style specified by ″style″ and that style has the same value as ″style,″ returns true, and ″starting″ and ″count″ represent the range over which that style has that value (even if other styles vary). If the styles at ″position″ don't include ″style,″ or ″style″ has a different value than the corresponding style in effect at ″position,″ returns false, and ″starting″ and ″count″ represent the range over which that style either doesn't exist or has a different value.

position
The position at which to check styles.

style The style to search for.

starting
If the return value is true, receives the starting offset of the range over which the specified style has the specified value. If the return value is false, receives the starting offset of the range over which the specified style does not have the specified value.

count If the return value is true, receives the length of the range over which the specified style has the specifid value. If the return value is false, receives the length of the range over which the specified style does not have the specified value.

Return
True if ″style″ matches one of the styles in effect at ″position″ in both class and value, false if it doesn't.

Exception

Exception	Description
IInvalidParameter	Thrown if position >= length().

sameCharacters

```
public:bool sameCharacters(const IText& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the characters in *this and ″other″ match exactly. (same as operator==()).

other A constant reference to the IText object to be compared with this one.

sameStyledText

```
public:bool sameStyledText(const IText& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if both the characters and styles in *this and ″other″ match exactly.

other A constant reference to the IText object to be compared with this one.

stylesAt

Returns style-run information for the specified position.

Overload 1

```
public:const ITextStyleSet*
  stylesAt( offset_type position,
            offset_type& starting,
            length_type& count ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

position
The position we're interested in getting style information for.

starting
Receives the starting offset of the style run containing the specified position.

count Receives the length of the style run containing the specified position.

Return
A pointer to a style set containing the styles in effect at the given position. If the text is unstyled at the given position, this set will be empty. ″starting″ and ″count″ represent the range over which the same styles (and no others) are applied to the text. This pointer is guaranteed to be good only until the next call to a non-const member of IText.

Exception

Exception	Description
IInvalidParameter	Thrown if position >= length().

Overload 2

```
public:void
  stylesAt( offset_type position,
            ITextStyleSet& set,
            offset_type& starting,
            length_type& count,
            ITextStyle::EStylePropagation propagationToMatch =
            ITextStyle::kMaximumPropagation ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this version of stylesAt() to narrow the function to consider only styles of a given propagation type. It also copies the styles into a fill-in parameter rather than returning a const pointer to a style set.

position
The position to get style information for.

set Receives the styles in effect at the specified position (″set″ is cleared first). If the caller specifies a propagation type, then ″set″ receives only the styles of the specified propagation type that are in effect at the given position.

starting
Receives the starting offset of the style run containing the specified position (i.e., the first offset where the styles of the given propagation type are the same as those in effect at the specified position).

count Receives the length of the style run containing the specified position (i.e., the length of the range over which the styles of the given propagation type are the same as those in effect at the specified position).

propagationToMatch
The propagation type we're interested in. Only styles of the given propagation type will be returned in "set," and only styles of the given propagation type will be considered in determining the values returned in "starting" and "count." If the caller passes ITextStyle::kMaximumPropagation, styles of all propagation types are considered.

Exception

Exception	Description
IInvalidParameter	Thrown if position >= length().

Inserting Characters

Use these member functions to insert one or more characters into the IText object.

insert

The original characters following the insertion position are moved to follow the new characters. It is legal to insert characters from an IText into itself. The new characters retain their character styles (raw UCS2Chars are considered unstyled), and paragraph styles are propagated.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText&
  insert( offset_type insertPos,
          const IText& that,
          offset_type thatPos,
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts characters into the IText object.

insertPos
The character offset where the first character of inserted text should go. A value equal to length() appends characters at the end.

that An IText containing the text to insert (both character and styles will be copied). May be equal to *this.

thatPos
The offset of the first character from "that" to insert.

thatLength
The number of characters from "that" to insert, or npos, which specifies all characters in "that" from thatPos to the end.

Exception

Exception	Description
IInvalidRequest	Thrown if the resulting length would be bigger then max_size().
IInvalidParameter	Thrown if thatPos > that.length().
IInvalidParameter	Thrown if insertPos > length().

Overload 2

```
public:void
  insert( ITextIterator insertPos,
          ITextIterator thatFrom,
          ITextIterator thatTo )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts characters into the IText.

insertPos
An iterator referring to the insertion position.

thatFrom
An iterator referring to the first character in the source string.

thatTo An iterator referring to the position AFTER last character in the source string.

Exception

Exception	Description
IInvalidParameter	Thrown if insertPos isn't a valid iterator on *this.
IInvalidParameter	Thrown if thatFrom and thatTo do not form a proper range.

Overload 3

```
public:IText&
  insert( offset_type insertPos,
          const ioc::unichar_t* that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IText&
  insert( offset_type insertPos,
          length_type thatCopies,
          ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IText&
  insert( offset_type insertPos,
          const ioc::unichar_t that [ ],
          length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:void
  insert( ITextIterator insertPos,
          length_type thatCopies,
          ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:ITextIterator
  insert( ITextIterator insertPos,
          ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IText& insert(offset_type insertPos, const IText& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts characters into the IText object.

insertPos
The character offset where the first character of inserted text should go. A value equal to length() appends characters at the end.

that The text to insert (both the characters and styles are copied). May be equal to *this.

Exception

Exception	Description
IInvalidRequest	Thrown if the resulting length would be bigger then max_size().
IInvalidParameter	Thrown if insertPos > length().

insert_and_propagate_styles

Inserts characters into the IText, and propagates styles over them. Same as the corresponding version of insert(), except that instead of going in as unstyled, the new characters take on the styles of the character preceding them (or, if they're inserted at the very beginning, the styles of the character following them).

Overload 1

```
public:IText&
  insert_and_propagate_styles( offset_type insertPos,
                               length_type thatCopies,
                               ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IText&
  insert_and_propagate_styles( offset_type insertPos,
                               const ioc::unichar_t that [ ],
                               length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

operator new

Overload 1

```
public:void* operator new(ioc::isize_t)
```

This is supported on **z/OS**

Overload 2

```
public:void* operator new(ioc::isize_t size, void* ptr)
```

This is supported on **z/OS**

Obtaining Information About the IText Object

Use these member functions to obtain information about the IText object such as its size, its maximum capacity, and the number of characters it currently contains.

c_str

```
public:const ioc::unichar_t* c_str() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to a null-terminated array of ioc::unichar_t data containing the same characters as contained by the IText. This pointer is guaranteed to be valid only until a subsequent call to c_str(), data(), or any non-const member function on the same IText. The calling code must not alter the contents of the array returned by c_str().

capacity

```
public:length_type capacity() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of characters the IText currently has space allocated for.

codeSet

```
public:IString codeSet() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character code set with which the IText object will transcode its text. If the IText object is not associated with a particular character code set, the character code set of the current locale will be used (retrieved from nl_langinfo(CODESET)).

data

```
public:const ioc::unichar_t* data() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as c_str(), but the array returned is not required to be null-terminated. A valid pointer is required to be returned even when the IText is empty, however. The calling code must not ater the contents of the array returned by data(). (The default implementation of IText will return the same thing for both c_str() and data().)

empty

Returns true if this->length() == 0.

```
public:bool empty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this->length() == 0.

length

```
public:length_type length() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as size().

max_size

```
public:length_type max_size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum number of characters an IText can hold. Currently, this will always be ULONG_MAX - 1.

reserve

```
public:void reserve(length_type numChars)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Acts as a signal to the IText that characters are about to be added. IText can use this hint as a signal to preallocate space for the new characters. The preallocation is not actually guaranteed to happen, however. If it does, capacity() will return a value greater than or equal to the value passed to reserve(). Otherwise, nothing about the IText is changed. reserve() cannot be used to make the allocated storage of the IText smaller. If numChars is less than capacity(), nothing happens. (The default implementation of reserve() will always allocate the extra memory, except in serious low-memory conditions.)

numChars

The number of characters for which space is reserved.

resize

Sets the size of the IText to a specified number. (i.e., after the user calls resize(), calls to size() will return the value passed to resize()). If the new size is smaller than the IText's current size, characters at the end of the IText are deleted. If the new size is larger than the IText's current size, an appropriate number of copies of the specified pad character are added at the end.

Overload 1

```
public:void resize(length_type numChars)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as calling the other overload with fillChar == UUnicodeSpecial::kNull.

numChars

The size for the IText object.

Overload 2

```
public:void resize(length_type numChars, ioc::unichar_t fillChar)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

size

```
public:length_type size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of characters in the IText.

storage_chunk

```
public:const ioc::unichar_t*
  storage_chunk( offset_type searchPos,
                 offset_type& chunkPos,
                 length_type& chunkLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the block of character storage containing the specified position. The pointer returned by this function is not guaranteed to be good after the next call to c_str(), data(), or a non-const member of this IText.

searchPos
The character offset of interest.

chunkPos
Receives the character offset of the first character in the storage chunk returned by this function. (output parameter only)

chunkLength
Receives the number of characters in the storage chunk returned. (output parameter only)

Return
A pointer to the block of character storage containing searchPos. This pointer is not guaranteed to be null-terminated.

Exception

Exception	Description
IInvalidParameter	Thrown if searchPos > length().

Replacing Characters

Use these member functions to replace characters in the IText object with those of the given entity.

replace

Replaces characters in an IText object.

If the character code set of *this is different from that (IText), the character code set information of *this is discarded. Otherwise, *this inherits the character code set from that (IText).

The character code set of *this is not changed if the parameter that is not an IText object.

Overload 1

```
public:IText&
  replace( ITextIterator thisFrom,
           ITextIterator thisTo,
           const IText& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces characters in an IText with characters from another IText.

Use this function to replace characters in an IText with characters from a null-terminated Unicode string If the original and replacement strings are not the same length, characters are inserted or deleted as necessary to make sure the characters on either side of the text being replaced are not disturbed. It is legal to replace characters in an IText with characters from the same IText, even if the

thisFrom
An iterator referring to first character to replace.

thisTo An iterator referring to first character AFTER the last one to be replaced.

that The new text (both characters and styles are copied). May be the same as *this.

Exception

Exception	Description
IInvalidParameter	Thrown if thisFrom and thisTo don't form a proper range on *this.
IInvalidRequest	Thrown if the resulting string would have a length greater than max_size().

Overload 2

```
public:IText&
  replace( offset_type thisPos,
           length_type thisLength,
           const IText& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces characters in an IText with the contents of another IText.

thisPos

The character offset of first character to replace.

thisLength

The number of characters to replace (npos may be used to say ″replace all characters from thisPos to the end″).

that The new text (both characters and styles are copied). May be the same as *this.

Exception

Exception	Description
IInvalidParameter	Thrown if thisPos > this->length().
IInvalidRequest	Thrown if the resulting string would have a length greater than max_size().

Overload 3

```
public:IText&
  replace( offset_type thisPos,
           length_type thisLength,
           length_type thatCopies,
           ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IText&
  replace( offset_type thisPos,
           length_type thisLength,
           const ioc::unichar_t* that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IText&
  replace( ITextIterator thisFrom,
           ITextIterator thisTo,
           const ioc::unichar_t that [ ],
           length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IText&
  replace( offset_type thisPos,
           length_type thisLength,
           const ioc::unichar_t that [ ],
           length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:IText&
  replace( ITextIterator thisFrom,
           ITextIterator thisTo,
           length_type thatCopies,
           ioc::unichar_t that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 8

```
public:IText&
  replace( ITextIterator thisFrom,
           ITextIterator thisTo,
           const ioc::unichar_t* that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 9

```
public:IText&
  replace( ITextIterator thisFrom,
           ITextIterator thisTo,
           ITextIterator thatFrom,
           ITextIterator thatTo )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces characters in an IText with characters from another IText.

thisFrom
An iterator referring to first character to replace.

thisTo An iterator referring to first character AFTER the last one to be replaced.

thatFrom
An iterator referring to first character to use in replacement string.

thatTo An iterator referring to position after the last character to use in replacement string.

Exception

Exception	Description
IInvalidParameter	Thrown if thisFrom and thisTo don't form a proper range on *this.
IInvalidParameter	Thrown if thatFrom and thatTo don't form a proper range.
IInvalidRequest	Thrown if the resulting string would have a length greater than max_size().

Overload 10

```
public:IText&
  replace( offset_type thisPos,
           length_type thisLength,
           const IText& that,
           offset_type thatPos,
           length_type thatLength )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces characters in an IText with characters from another IText.

thisPos
The character offset of first character to replace.

thisLength
The number of characters to replace (npos may be used to say ″replace all characters from thisPos to the end″).

that The IText containing the new text (both characters and styles are copied). May be the same as *this.

thatPos
The offset of first character in ″that″ to use as part of the replacement string.

thatLength
The number of characters from ″that″ to use as replacement string.

Exception

Exception	Description
IInvalidParameter	Thrown if thisPos > this->length() or thatPos > that.length().
IInvalidRequest	Thrown if the resulting string would have a length greater than max_size().

Searching the IText Object

Use these member functions to search the IText object for one or more characters.

find

This is a bitwise search, and styles are ignored.

Overload 1

```
public:offset_type
  find( ioc::unichar_t key,
        offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:offset_type
  find( const ioc::unichar_t key [ ],
        offset_type startPos,
        length_type keyLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  find( const ioc::unichar_t* key,
        offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  find( const IText& key,
        offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches the IText for the specified string.

key String to search for

startPos

position of first character to examine in the search

Return

offset of the first character of the first instance of ″key″ found in the IText at or after startPos, or npos if ″key″ is not found

Exception

Exception	Description
IInvalidParameter	Thrown if startPos > length().

find_first_not_of

Searches the IText for characters not in the specified set or characters that don't match the specified one. (This is a bitwise search, and styles are ignored.)

Overload 1

```
public:offset_type
  find_first_not_of( const IText& keys,
                     offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches the IText for characters not in the specified set.

keys An IText containing the characters to search for.

startPos

Position of first character to examine in the search.

Return

offset of the first character in the IText at or after startPos that does not occur in ″keys,″ or npos if no characters that aren't in ″keys″ are found

Exception

Exception	Description
IInvalidParameter	Thrown if startPos > length().

Overload 2

```
public:offset_type
  find_first_not_of( const ioc::unichar_t keys [ ],
                     offset_type startPos,
                     length_type numKeys ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  find_first_not_of( const ioc::unichar_t* keys,
                     offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  find_first_not_of( ioc::unichar_t key,
                     offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

find_first_of

Searches the IText for the specified character or set of characters. (This is a bitwise search, and styles are ignored.)

Overload 1

```
public:offset_type
  find_first_of( const IText& keys,
                 offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches the IText for the specified characters.

keys An IText containing the characters to search for.

startPos

Position of first character to examine in the search.

Return

offset of the first character in the IText at or after startPos that occurs in ″keys,″ or npos if no characters in ″keys″ are found

Exception

Exception	Description
IInvalidParameter	Thrown if startPos > length().

Overload 2

```
public:offset_type
  find_first_of( const ioc::unichar_t* keys,
                 offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  find_first_of( ioc::unichar_t key,
                 offset_type startPos = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  find_first_of( const ioc::unichar_t keys [ ],
                 offset_type startPos,
                 length_type numKeys ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

find_last_not_of

Searches backwards through the IText for characters not in the specified set. (This is a bitwise search, and styles are ignored.)

Overload 1

```
public:offset_type
  find_last_not_of( const ioc::unichar_t keys [ ],
                    offset_type startPos,
                    length_type numKeys ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:offset_type
  find_last_not_of( ioc::unichar_t key,
                    offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  find_last_not_of( const ioc::unichar_t* keys,
                    offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  find_last_not_of( const IText& keys,
                    offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches the IText for characters not in the specified set.

keys An IText containing the characters to search for.

startPos

Position AFTER the first character to examine in the search (npos may be used to tell the function to search the whole string).

Return

offset of the last character in the IText before startPos that does not occur in ″keys,″ or npos if no characters that aren't in ″keys″ are found

find_last_of

Searches backwards through the IText for the specified character or set of characters. (This is a bitwise search, and styles are ignored.)

Overload 1

```
public:offset_type
  find_last_of( ioc::unichar_t key,
                offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:offset_type
  find_last_of( const ioc::unichar_t keys [ ],
                offset_type startPos,
                length_type numKeys ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  find_last_of( const ioc::unichar_t* keys,
                offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  find_last_of( const IText& keys,
                offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches backwards through the IText for the specified characters.
keys An IText containing the characters to search for.
startPos
Position AFTER the first character to examine in the search (npos may be used to tell the function to search the whole string).
Return
offset of the last character in the IText before startPos that occurs in ″keys,″ or npos if no characters in ″keys″ are found

rfind

Searches backwards through the IText for the specified character or string. (This is a bitwise search, and styles are ignored.)

Overload 1

```
public:offset_type
  rfind( const ioc::unichar_t* key,
         offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:offset_type
  rfind( const ioc::unichar_t key [ ],
         offset_type startPos,
         length_type keyLength ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:offset_type
  rfind( ioc::unichar_t key,
         offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:offset_type
  rfind( const IText& key,
         offset_type startPos = npos ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Searches backwards through the IText for the specified string.
key String to search for

startPos

position AFTER the first character to examine in the search (npos may be used to tell the function to search the whole string)

Return

offset of the first character of the last instance of ″key″ found in the IText before startPos, or npos if ″key″ is not found

Streaming the IText Object In and Out

Use these operators to write the IText object out to the specified stream and read it in from it.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reads the contents of the IText (both characters and styles) from fromWhere. The text read from the stream replaces any text already in the IText. This function depends on the data being in the same stream format as written by operator>>=().

fromWhere

The stream from which the IText object is read in.

Return

fromWhere.

operator >>=

```
public:void operator >>=(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the contents of the IText (both characters and styles) to toWhere. The streaming is optimized: the character and style data are both compressed, the style data takes up zero bytes when the IText is unstyled, and overhead is kept to a bare minimum.

toWhere

The stream to which the IText object is written out.

Return

toWhere.

Typedefs and Constant Definitions

Type definitions and constants defined by this class.

npos

```
public:static const length_type npos
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

″To the end″ length, or ″not found″ flag. This is a special value that is used in the interface to IText for two purposes:

1. In length parameters, specifies all characters between the specified starting offset and the end of the string. (A starting offset of 0 and a length of npos refers to the whole string.)
2. Returned by the find functions when they fail to find the search string.

npos is equal to (unsigned long)(-1). For example, the largest possible unsigned long.

IText - Type Definitions

length_type

```
typedef ioc::isize_t length_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Type used for specifying character counts. Always typedefed to size_t because the ANSI standard uses size_t. Always represents character counts, not byte counts.

offset_type

```
typedef ioc::isize_t offset_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Type used for specifying character offsets. Always typedefed to size_t because the ANSI standard uses size_t. All offset_type parameters are zero-based: 0 refers to the position of the first character in a string, and length() returns the offset of the first character position beyond the end of the string. There is no concept of ″insertion position″ (i.e., numbering the ″cracks″ between the characters rather than the characters themselves) in the standard, as there was in the old Standard Text library; by convention, insertions take place before the character whose offset is specified as an insertion position (which has the same effective result as numbering the cracks between characters). You add at the end by specifying the past-the-end offset (i.e., the value returned by length()) as the insertion position.

IText - Inherited Member Functions and Data

Inherited Public Functions

IText			
Definition	**Page Number**	**Definition**	**Page Number**
`emptyText`	560		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextBitmapStyle

The ITextBitmapStyle class provides a style that controls whether the text it is applied to is displayed with bitmap or outline fonts. TRUE means use bitmap fonts.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextBitmapStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextBitmapStyle

ITextBitmapStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextBitmapStyle objects.

~ITextBitmapStyle

```
public:virtual ~ITextBitmapStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextBoldfaceStyle object.

ITextBitmapStyle

Constructs an ITextBitmapStyle object.

Overload 1

```
public:ITextBitmapStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextBitmapStyle object that is initialized to use bitmap fonts.
isSet The initialization value, equivalent to TRUE.

Overload 2

```
public:ITextBitmapStyle(const ITextBitmapStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextBitmapStyle object that is constructed from the given one.
other The ITextBitmapStyle object to be copied.

Overload 3

```
public:ITextBitmapStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextBitmapStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Assignment Operator

Use this operator to assign the given ITextBitmapStyle object to the target one.

operator =

```
public:ITextBitmapStyle& operator =(const ITextBitmapStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.
other The replacement ITextBitmapStyle object.

Comparing ITextBitmapStyle Objects

Use these operators to compare the given ITextBitmapStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextBitmapStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextBitmapStyle object against the current one for inequality.

other The ITextBitmapStyle object to be compared.

operator ==

```
public:bool operator ==(const ITextBitmapStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextBitmapStyle object against the current one for equality.

other The ITextBitmapStyle object to be compared.

Getting and Setting the Bitmap Font Value

Use these member functions to get and set the value that specifies the font to be used to render the text.

isBitmap

```
public:bool isBitmap() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether this is a bitmap font or an outline one. A returned value of TRUE indicates that it is a bitmap.

setBitmap

```
public:void setBitmap(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the flag that determines whether the text the style applies to should be drawn using a bitmap font or an outline font (true means bitmap).

value The flag value. TRUE means use the bitmap font, false means use the outline font.

Protected Members

addToSet

```
protected:virtual void addToSet(ITextStyleSet& set) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

getValueFromSet

```
protected:virtual void getValueFromSet(const ITextStyleSet& set)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isContainedInSet

```
protected:virtual bool
  isContainedInSet( const ITextStyleSet& set ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isValueContainedInSet

```
protected:virtual bool
  isValueContainedInSet( const ITextStyleSet& set ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

removeFromSet

```
protected:virtual void removeFromSet(ITextStyleSet& set) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITextBitmapStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextBoldfaceStyle

The ITextBoldfaceStyle class provides a style that determines whether the text it is applied to is boldface or lightface. TRUE means bold.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextBoldfaceStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextBoldfaceStyle

ITextBoldfaceStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextBoldfaceStyle objects.

~ITextBoldfaceStyle

```
public:virtual ~ITextBoldfaceStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextBoldfaceStyle object.

ITextBoldfaceStyle

Constructs an ITextBoldfaceStyle object.

Overload 1

```
public:ITextBoldfaceStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextBoldfaceStyle object whose value is initialized TRUE, indicating use of boldface.

Overload 2

```
public:ITextBoldfaceStyle(const ITextBoldfaceStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextBoldfaceStyle object that is constructed from the given one.

other The ITextBoldfaceStyle object to be copied.

Overload 3

```
public:ITextBoldfaceStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextBoldfaceStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Assignment Operator

Use this operator to assign the given ITextBoldfaceStyle object to the target one.

operator =

```
public:ITextBoldfaceStyle&
  operator =( const ITextBoldfaceStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The replacement ITextBoldfaceStyle object.

Comparing ITextBoldfaceStyle Objects

Use these operators to compare the given ITextBoldfaceStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextBoldfaceStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextBoldfaceStyle object against the current one for inequality.

other The ITextBoldfaceStyle object to be compared.

operator ==

```
public:bool operator ==(const ITextBoldfaceStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextBoldfaceStyle object against the current one for equality.

other The ITextBoldfaceStyle object to be compared.

Getting and Setting the Boldface Flag

Use these member functions to get and set the value of the flag that determines whether boldface or lightface is used to render the text.

isBoldface

```
public:bool isBoldface() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether boldface or lightface is applied to the text. A return value of TRUE indicates boldface.

setBoldface

```
public:void setBoldface(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the flag that determines whether boldface or lightface is to be applied to the text.

value The flag setting. TRUE specifies boldface.

ITextBoldfaceStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextBoldfaceStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	598	getValueFromSet	598
isContainedInSet	598	isValueContainedInSet	598
removeFromSet	598		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextBoundary

The ITextBoundary class implements methods for finding the location of boundaries in text. ITextBoundary is an abstract base class. Instances of ITextBoundary maintain a current position and scan over text returning the index of characters where boundaries occur.

Character boundary analysis allows users to interact with characters as they expect to, for example, when moving the cursor through a text string. Character boundary analysis provides correct navigation through strings regardless of how a character is represented. For example, an accented character might be stored as a base character and a diacritical mark, or a single combined character. What users consider to be a character can differ between languages.

Word boundary analysis is used by search and replace functions, as well as within text editing applications that allow the user to select words with a double click. Word selection provides correct interpretation of punctuation marks within and following words. Characters that are not part of a word, such as symbols or punctuation marks, have word breaks on both sides.

Line boundary analysis determines where a text string can be broken when line-wrapping. The mechanism correctly handles punctuation and hyphenated words.

Sentence boundary analysis allows selection with correct interpretation of periods within numbers and abbreviations, and trailing punctuation marks such as quotation marks and parentheses.

This is the interface for all text boundaries.

Examples:

Print each word of the text in order

```
void printEachWordForward(const IText& text) {
    ITextBoundary* wordBoundary = ITextBoundary::createWordBreak();
    words->setText(text);
    for (ITextBoundary::offset_type start = wordBoundary->first(),
         end = wordBoundary->next();
         end != ITextBoundary::kDone;
         start = end, end = wordBoundary->next())
    {
      printf(text.substr(start, end - start));
    }
    delete wordBoundary;
}
```

Print each line of the text in reverse order

```
void printEachLineBackward(const IText& text) {
    ITextBoundary* lineBoundary = ITextBoundary::createLineBreak();
    lineBoundary->setText(text);
    for (ITextBoundary::offset_type end = lineBoundary->last(),
         start = lineBoundary->previous();
         start != ITextBoundary::kDone;
         end = start, start = lineBoundary->previous())
    {
        printf(text.substr(start, end - start));
    }
    delete lineBoundary;
}
```

Class header file: itxtbdry.hpp

ITextBoundary - Hierarchy List

ITextBoundary

ITextBoundary - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextBoundary objects.

~ITextBoundary

```
public:virtual ~ITextBoundary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextBoundary

```
protected:ITextBoundary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ITextBoundary object.

The default constructor.

Comparing Two ITextBoundary Objects for Equality and Inequality

Use these operators to compare the contents of the given ITextBoundary object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextBoundary& rhs) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the negation of operator==.

rhs A constant reference to the ITextBoundary object to be compared with the current one.

operator ==

```
public:virtual bool operator ==(const ITextBoundary&) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this ITextBoundary object is at the same position in the same text and is the same class and type (word, line, etc.) of ITextBoundary, as the argument. Text is considered the same if it contains the same characters, it need not be the same object, and styles are not considered.

ITextBoundary

A constant reference to the ITextBoundary object to be compared with the current one.

Getting a Polymorphic Copy

Use this member function to obtain a polymorphic copy of the ITextBoundary object.

clone

```
public:virtual ITextBoundary* clone() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a polymorphic copy of this ITextBoundary.

Getting and Setting the Text

Use these member functions to get and set the text operated on.

setText

```
public:virtual void setText(const IText&) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The text boundary is reset to the start.

IText A constant reference to the IText object containing the text.

text

```
public:virtual IText text() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the text over which this operates.

Getting Text Boundary Indexes

Use these member functions to get indexes to various text boundaries.

current

```
public:virtual offset_type current() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character index of the text boundary that was most recently returned by next(), previous(), first(), last(), or nthFromCurrent().

first

```
public:virtual offset_type first() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the index of the first character in the text being scanned.

last

```
public:virtual offset_type last() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the index following the last character in the text being scanned.

next

```
public:virtual offset_type next() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character index of the next text boundary, or kDone if all boundaries have been returned.

nextAfter

```
public:virtual offset_type nextAfter(offset_type offset) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The value returned is always greater than the offset, or is kDone.

offset The offset to begin scanning.

Return

The first boundary after the specified offset.

nthFromCurrent

```
public:virtual offset_type nthFromCurrent(int n) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the nth boundary from the current boundary.

n The signed number of boundaries to traverse. Negative values move to previous boundaries and positive values move to later boundaries. A value of 0 does nothing. This function does affect the internal interation state of the text boundary, so if you call current() after calling nthFromCurrent, current() will return the same value nthFromCurrent just did; and repeated calls to nthFromCurrent will advance through the text.

previous

```
public:virtual offset_type previous() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the character index of the previous text boundary, or kDone if all boundaries have been returned.

Getting the User-Displayable Name

Use this member function to get the user-displayable name in the locale's language.

getDisplayName

```
public:static IText
  getDisplayName( const ILocaleKey& objectLocale = ILocaleKey::defaultLocale ( ),
                  const ILocaleKey& displayLocale = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets a user-displayable name in the language of the display locale that refers to ITextBoundary objects for the object locale.

objectLocale
A locale used to obtain an ITextBoundary, as used in createWordBreak, createLineBreak, and so forth.

displayLocale
The locale identifying the language to use for the returned name.

Locating Line, Word, and Sentence Breaks

Use these member functions to create instances of ITextBoundary objects that locate sentence, line, and word breaks.

createCharacterBreak

```
public:static ITextBoundary*
  createCharacterBreak( const ILocaleKey& locale = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Character breaks are boundaries of combining character sequences. These are useful for determining possible places for caret placement or cursor movement.

locale The locale. If an ITextBoundary for character breaks is not avaliable for the specified locale, a default ITextBoundary for character breaks is returned.

Return
A new instance of ITextBoundary that locates character breaks

createLineBreak

```
public:static ITextBoundary*
  createLineBreak( const ILocaleKey& locale = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Line breaks are useful for locating places where a line may be broken when flowing text into a container. This locates all logically possible line breaks. Clients can select from these to determine which line break to use based on a maximum line width, for example.

locale The locale. If an ITextBoundary for line breaks is not avaliable for the specified locale, a default ITextBoundary for line breaks is returned.

Return

A new instance of ITextBoundary that locates line breaks

createSentenceBreak

```
public:static ITextBoundary*
  createSentenceBreak( const ILocaleKey& locale = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sentence breaks are useful for sentence selection.

locale The locale. If an ITextBoundary for sentence breaks is not avaliable for the specified locale, a default ITextBoundary for sentence breaks is returned.

Return

A new instance of ITextBoundary that locates sentence breaks

createWordBreak

```
public:static ITextBoundary*
  createWordBreak( const ILocaleKey& locale = ILocaleKey::defaultLocale ( ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Word breaks are useful for word selection (for example, when double clicking with the mouse).

locale The locale. If an ITextBoundary for word breaks is not avaliable for the specified locale, a default ITextBoundary for word breaks is returned.

Return

A new instance of ITextBoundary that locates word breaks

Miscellaneous Members

kDone

```
public:static offset_type kDone
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This value is returned by previous() and next() after all valid boundaries have been returned.

ITextBoundary - Type Definitions

offset_type

```
typedef IText::offset_type offset_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A convenient name for the type representing offsets into the text.

ITextBoundary - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextColorStyle

The ITextColorStyle class provides a style that specifies the color of the text it is applied to. The color is specified as three separate RGB components, each an unsigned char that can range in value from 0 to 255.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextColorStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextColorStyle

ITextColorStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextColorStyle objects.

~ITextColorStyle

```
public:virtual ~ITextColorStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextColorStyle

Constructs an ITextColorStyle object.

Overload 1

```
public:ITextColorStyle( unsigned char red,
                        unsigned char green,
                        unsigned char blue )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextColorStyle object from the given red, green, and blue values.

red The red value to be used for the new object.
green The green value to be used for the new object.
blue The blue value to be used for the new object.

Overload 2

```
public:ITextColorStyle(const ITextColorStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of class ITextColorStyle that is copied from the given one.
other The ITextColorStyle object that is copied.

Overload 3

```
public:ITextColorStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextColorStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Assignment Operator

Use this operator to assign the given ITextColorStyle object to the target one.

operator =

```
public:ITextColorStyle& operator =(const ITextColorStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the given ITextColorStyle object with the target one.
other The ITextColorStyle object to be assigned.

Comparing ITextColorStyle Objects

Use these operators to compare the given ITextColorStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextColorStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextColorStyle object with the current one for inequality.
other The ITextColorStyle object to be compared with the current one.

operator ==

```
public:bool operator ==(const ITextColorStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextColorStyle object with the current one for equality.
other The ITextColorStyle object to be compared with the current one.

Getting and Setting the RGB Values

Use these member functions to get and set the red, green, and blue color values either collectively or individually.

blue

```
public:unsigned char blue() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the blue value of the RGB.

colors

Gets the red, green, and blue color values.

```
public:void
  colors( unsigned char& red,
          unsigned char& green,
          unsigned char& blue ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets all three RGB values.
red The red value.
green The green value.
blue The blue value.

green

```
public:unsigned char green() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the green value of the RGB.

red

```
public:unsigned char red() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the red value of the RGB.

setBlue

```
public:void setBlue(unsigned char newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the blue value of the RGB.
newValue
The blue value to be used as the new setting.

setColors

```
public:void
  setColors( unsigned char newRed,
             unsigned char newGreen,
             unsigned char newBlue )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets all three RGB color values.
newRed
The red value to be used as the new setting.
newGreen
The green value to be used as the new setting.
newBlue
The blue value to be used as the new setting.

setGreen

```
public:void setGreen(unsigned char newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the green value of the RGB.
newValue
The green value to be used as the new setting.

setRed

```
public:void setRed(unsigned char newValue)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the red value of the RGB.

newValue

The red value to be used as the new setting.

ITextColorStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextColorStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	606	getValueFromSet	606
isContainedInSet	606	isValueContainedInSet	606
removeFromSet	606		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextFixedPitchStyle

The ITextFixedPitchStyle class provides a style that determines whether the text it is applied to is fixed pitch. TRUE means fixed pitch.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextFixedPitchStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextFixedPitchStyle

ITextFixedPitchStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextFixedPitchStyle objects.

~ITextFixedPitchStyle

```
public:virtual ~ITextFixedPitchStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextFixedPitchStyle object.

ITextFixedPitchStyle

Constructs an ITextFixedPitchStyle object.

Overload 1

```
public:ITextFixedPitchStyle(const ITextFixedPitchStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor.

other The ITextFixedPitchStyle object that is copied.

Overload 2

```
public:ITextFixedPitchStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextFixedPitchStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Overload 3

```
public:ITextFixedPitchStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of class ITextFixedPitchStyle using the given boolean value to set the flag that determines whether fixed pitch font is used.

isSet The style flag setting to be used in constructing the ITextFixedPitchStyle object.

Assignment Operator

Use this operator to assign the given ITextFixedPitchStyle object to the target one.

operator =

```
public:ITextFixedPitchStyle&
  operator =( const ITextFixedPitchStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The replacement ITextFixedPitchStyle object.

Comparing ITextFixedPitchStyle Objects

Use these operators to test the given ITextFixedPitchStyle object against the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextFixedPitchStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextFixedPitchStyle object against the current one for inequality.

other The ITextFixedPitchStyle object to be compared.

operator ==

```
public:bool operator ==(const ITextFixedPitchStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextFixedPitchStyle object against the current one for equality.

other The ITextFixedPitchStyle object to be compared.

Getting and Setting the FixedPitch Flag

Use these member functions to get and set the value of the flag that determines whether or not this is a fixed pitch font.

isFixedPitch

```
public:bool isFixedPitch() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether this is a fixed pitch font.

setFixedPitch

```
public:void setFixedPitch(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the flag that determines whether this is a fixed pitch font.

value The flag setting. TRUE specifies fixed pitch.

ITextFixedPitchStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextFixedPitchStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	610	getValueFromSet	610
isContainedInSet	610	isValueContainedInSet	610
removeFromSet	610		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextItalicStyle

The ITextItalicStyle class provides a style setting that determines whether the text the style is applied to is rendered in italic or roman. TRUE means italic.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextItalicStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextItalicStyle

ITextItalicStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextItalicStyle objects.

~ITextItalicStyle

```
public:virtual ~ITextItalicStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextItalicStyle object.

ITextItalicStyle

Constructs an ITextItalicStyle object.

Overload 1

```
public:ITextItalicStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an object of class ITextItalicStyle using the given boolean value to set the flag that determines whether Roman or Italic is used.

isSet The style flag setting to be used in constructing the ITextItalicStyle object.

Overload 2

```
public:ITextItalicStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextItalicStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Overload 3

```
public:ITextItalicStyle(const ITextItalicStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor.

other The ITextItalicStyle object that is copied.

Assignment Operator

Use this operator to assign the given ITextItalicStyle object to the target one.

operator =

```
public:ITextItalicStyle& operator =(const ITextItalicStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the given ITextItalicStyle object to the target one.

other The ITextItalicStyle object to be assigned.

Comparing ITextItalicStyle Objects

Use these operators to test two ITextItalicStyle objects for equality or inequality.

operator !=

```
public:bool operator !=(const ITextItalicStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextItalicStyle object against the current one for inequality.

other The ITextItalicStyle object to be tested against the current one.

operator ==

```
public:bool operator ==(const ITextItalicStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextItalicStyle object against the current one for equality.

other The ITextItalicStyle object to be tested against the current one.

Getting and Setting the Style Value

Use these member functions to set the ITextItalicStyle style flag that specifies whether Italic or Roman is to be used, and to get the current setting.

isItalic

```
public:bool isItalic() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether the style setting is Italic or Roman.

setItalic

```
public:void setItalic(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the style flag using the given value.

value A Boolean that sets the style value. TRUE means set it to use Italic.

ITextItalicStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextItalicStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet isContainedInSet removeFromSet	613 613 613	getValueFromSet isValueContainedInSet	613 613

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet isContainedInSet ITextStyle removeFromSet	633 633 632 633	getValueFromSet isValueContainedInSet operator =	633 633 633

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextIterator

The ITextIterator class is a forward iterator for IText. It provides read-write sequential access to the characters in an IText, and can be used as a pointer to an individual character in an IText.

This class obeys the protocol for an STL-compatible random-access iterator.

Do not derive your own classes from this class.

Class header file: itxtiter.hpp

ITextIterator - Hierarchy List

ITextIterator

ITextIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextIterator objects.

~ITextIterator

```
public:~ITextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextIterator

Constructs an ITextIterator object.

Overload 1

```
public:ITextIterator(const ITextIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextIterator that is copied from the given one.

Overload 2

```
public:ITextIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextIterator object that leaves the iterator in an undefined state. You must call the assignment operator before you can use a default-constructed iterator.

Assignment Operator

Use this operator to replace the current ITextIterator with the given one.

operator =

```
public:ITextIterator& operator =(const ITextIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Points this to the same character position in the same IText as that points to.

Comparing ITextIterator Objects

Use these operators to compare the given ITextIterator object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the iterators refer to different character positions in the same IText. This is a straight pointer comparison. No error checking is performed.

Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

operator ==

```
public:bool operator ==(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if both iterators refer to the same character position in the same IText. This is a straight pointer comparison. No error checking is performed.

Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

Determining the Character the Iterator Refers To

Use these operators to obtain the number of character positions between the two iterators or to compare the positions of the given iterator and the current one.

operator *

```
public:ICharacterReference operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the character the iterator refers to.

Exception

Exception	Description
IInvalidParameter	if the iterator refers to an out-of-range character position on its IText.

operator <

```
public:bool operator <(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this refers to a character with a lower offset than that does.

Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

operator <=

```
public:bool operator <=(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this refers to the same character as that, or to one with a lower offset.

Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

operator >

```
public:bool operator >(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this refers to a character with a higher offset than that does.
Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

operator >=

```
public:bool operator >=(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if this refers to the same character as that, or to one with a higher offset.
Exception

Exception	Description
IInvalidParameter	if both iterators don't refer to the same IText

operator []

```
public:ICharacterReference
  operator [](  ioc::numeric_t offset ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character that is the specified number of offsets, or positions, away from the one the iterator refers to. For example, if iter points to position 2 in a function, iter[3] will return the character at position 5 in that function, not the character at position 3. The iterator itself is left unchanged by this operation.

offset The number of offsets.

Identifying the IText Object of the Iterator

Use this member function to identify the IText object to which the iterator belongs and acts on.

text

```
public:IText* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the IText the iterator refers to.

Incrementing and Decrementing the Iterator

Use these operators to move the iterator, traversing characters.

operator ++

Advances the iterator to the next character position, that is, the position with the next highest offset. This function performs no error checking; rather, it depends on operator*() for error checking.

Overload 1

Advances the iterator to the next character position.

```
public:ITextIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

Advances the iterator to the next character position

```
public:ITextIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return

A copy of the iterator before it was advanced.

operator +=

```
public:ITextIterator& operator +=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator the number of character positions specified by distance. It advances the iterator toward the end of the text, that is, toward higher character offsets. This function performs no error checking; it depends on operator*() for error checking. Returns a reference to this.

distance

The number of character positions to advance the text.

operator -

```
public:ITextIterator operator -(ioc::numeric_t distance) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator --

Advances the iterator to the previous character position, that is, the one with the next-lower offset. This routine performs no error checking; it depends on operator*() for error checking.

Overload 1

Advances the iterator to the previous character position

```
public:ITextIterator operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the iterator before it was advanced.

Return

A copy of the iterator before it was advanced.

Overload 2

Advances the iterator to the previous character position

```
public:ITextIterator& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to this.

operator -=

```
public:ITextIterator& operator -=(ioc::numeric_t distance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Advances the iterator by the distance of character positions toward the beginning of the text, that is, toward lower character offsets. Returns a reference to this. This routine performs no error checking; it depends on operator*() for error checking.

distance

The number of character positions to advance the text.

Obtaining Information about the Characters Referred To

Use these member functions to obtain the number of characters in the range marked by the iterators or to obtain the offset of the character referred to.

offset

```
public:ioc::isize_t offset() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character offset within its IText of the character the iterator refers to.

operator -

Returns the number of character positions between the two iterators

```
public:ioc::numeric_t operator -(const ITextIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of character positions between the two iterators. This has the same effect as offset() - that.offset().

that The given iterator used with this one to mark the text range.

Exception

Exception	Description
IInvalidParameter	if the iterators refer to different ITexts

ITextIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextOutlineStyle

The ITextOutlineStyle class provides a style that determines whether the text it is applied to is drawn in outline or not. TRUE means outline.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextOutlineStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextOutlineStyle

ITextOutlineStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextOutlineStyle objects.

~ITextOutlineStyle

```
public:virtual ~ITextOutlineStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITextOutlineStyle object.

ITextOutlineStyle

Constructs an ITextOutlineStyle object.

Overload 1

```
public:ITextOutlineStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextOutlineStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Overload 2

```
public:ITextOutlineStyle(const ITextOutlineStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextOutlineStyle object that is copied from the given one.

other The ITextOutlineStyle object that is copied.

Overload 3

```
public:ITextOutlineStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextOutlineStyle object that is initialized to true.

isSet The flag setting to be used in constructing the ITextOutlineStyle object.

Assignment Operator

Use this operator to assign the given ITextOutlineStyle object to the target one.

operator =

```
public:ITextOutlineStyle&
  operator =( const ITextOutlineStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The replacement ITextOutlineStyle object.

Comparing ITextOutlineStyle Objects

Use these operators to compare the given ITextOutlineStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextOutlineStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextOutlineStyle object against the current one for inequality.
other The ITextOutlineStyle object tested with this, the current, one.

operator ==

```
public:bool operator ==(const ITextOutlineStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextOutlineStyle object against the current one for equality.
other The ITextOutlineStyle object tested with this, the current, one.

Getting and Setting the Outline Style

Use these member functions to get and set the current setting of the outline style flag. Setting the flag to TRUE specifies that the text be drawn in outline style.

isOutlined

```
public:bool isOutlined() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests whether the ITextOutlineStyle flag is set to specify outline font.

setOutlined

```
public:void setOutlined(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the ITextOutlineStyle flag.
value The value to use to set the flag.

ITextOutlineStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextOutlineStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	620	getValueFromSet	620
isContainedInSet	620	isValueContainedInSet	620
removeFromSet	620		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextPlacement

Class header file: itxtplmt.hpp

ITextPlacement - Hierarchy List

ITextPlacement

ITextPlacement - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextPlacement objects.

~ITextPlacement

```
public:~ITextPlacement()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITextPlacement object.

ITextPlacement

Constructs an ITextPlacement object.

Overload 1

```
public:ITextPlacement(const ITextPlacement& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextPlacement that is copied from the given one.

Overload 2

```
public:ITextPlacement( const ITextIterator& pos,
                       ITextPlacement::ECharPlacement placement )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextPlacement that is copied from the given ITextIterator object. The placement is specified by placement.

Overload 3

```
public:ITextPlacement(const ITextIterator& pos)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextPlacement that is copied from the given ITextIterator object. Default placement is AFTER_OFFSET.

Assignment Operator

Use this operator to assign the given ITextPlacement object to this, the current, one.

operator =

```
public:ITextPlacement& operator =(const ITextPlacement& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

source The ITextPlacement object that replaces the given one.

Compares ITextPlacement Objects

Use these operators to test the given ITextPlacement object with the current one to determine if their contents are the same or different, or to test them for equality or greater than or less than positions.

operator !=

```
public:bool operator !=(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextPlacement object against the current one to determine if they are the different.

other The ITextPlacement object to be compared with this, the current, one.

operator <

```
public:bool operator <(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if fPos < other.fPos or if they're equal but fPlacement is BEFORE_OFFSET and other.fPlacement is AFTER_OFFSET

operator <=

```
public:bool operator <=(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns (*this < other || *this == other).

operator ==

```
public:bool operator ==(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextPlacement object against the current one to determine if they are the same.

other The ITextPlacement object to be compared with this, the current, one.

operator >

```
public:bool operator >(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if fPos > other.fPos or if they're equal but fPlacement is AFTER_OFFSET and other.fPlacement is BEFORE_OFFSET

operator >=

```
public:bool operator >=(const ITextPlacement& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns (*this > other || *this == other).

Modifying the Text Placement

Use these operators to increment or decrement the text placement.

operator ++

If fPlacement is BEFORE_OFFSET, flips it to AFTER_OFFSET; otherwise, increments fPos and sets fPlacement to BEFORE_OFFSET.

Overload 1

```
public:ITextPlacement& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITextPlacement& operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator --

If fPlacement is AFTER_OFFSET, flips it to BEFORE_OFFSET; otherwise, decrements fPos and sets fPlacement to AFTER_OFFSET.

Overload 1

```
public:ITextPlacement& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITextPlacement& operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

fPlacement

```
public:ECharPlacement fPlacement
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A constant that refers to the character placement.

fPos

```
public:ITextIterator fPos
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A constant that refers to the text iterator.

ITextPlacement - Enumerations

ECharPlacement

Specifies positioning bias:

- BEFORE_OFFSET - Insertion point is drawn after the character that precedes the character at fPos.
- AFTER_OFFSET - Insertion point is drawn before the character at fPos.

ITextPlacement - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextPointSizeStyle

The ITextPointSizeStyle class controls the point size of a range of text. The point size is specified as a float.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextPointSizeStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextPointSizeStyle

ITextPointSizeStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextPointSizeStyle objects.

~ITextPointSizeStyle

```
public:virtual ~ITextPointSizeStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextPointSizeStyle

Contructs an ITextPointSizeStyle object.

Overload 1

```
public:ITextPointSizeStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an ITextPointSizeStyle object that leaves the style in an undefined state. You must fill in the default-constructed style before you use it.

Overload 2

```
public:ITextPointSizeStyle(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextPointSizeStyle object that is initialized to the given floating point value.
value The point size to be set for the object.

Overload 3

```
public:ITextPointSizeStyle(const ITextPointSizeStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextPointSizeStyle object that is copied from the given one.
other The ITextPointSizeStyle object that is copied.

Assignment Operator

Use this operator to assign the given ITextPointSizeStyle object to the target one.

operator =

```
public:ITextPointSizeStyle&
  operator =( const ITextPointSizeStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.
other The ITextPointSizeStyle object to be assigned.

Comparing ITextPointSizeStyle Objects

Use these operators to compare the given ITextPointSizeStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextPointSizeStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextPointSizeStyle object against the current one for inequality.
other The ITextPointSizeStyle object to test against the current one.

operator ==

```
public:bool operator ==(const ITextPointSizeStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextPointSizeStyle object against the current one for equality.
other The ITextPointSizeStyle object to test against the current one.

Getting and Setting the Point Size

Use these member functions to get and set the point size.

pointSize

```
public:float pointSize() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current setting of the point size.

setPointSize

```
public:void setPointSize(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the point size.
value A floating point value that specifies the point size.

ITextPointSizeStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextPointSizeStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	626	getValueFromSet	626
isContainedInSet	626	isValueContainedInSet	626
removeFromSet	626		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextStrikethroughStyle

The ITextStrikethroughStyle class provides a style that determines whether the text that the style is applied to will have a strikethrough bar or not. TRUE means strikethrough is used.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextStrikethroughStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextStrikethroughStyle

ITextStrikethroughStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextStrikethroughStyle objects.

~ITextStrikethroughStyle

```
public:virtual ~ITextStrikethroughStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextStrikethroughStyle object.

ITextStrikethroughStyle

Constructs an ITextStrikethroughStyle object.

Overload 1

```
public:ITextStrikethroughStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextStrikethroughStyle object that is initialized to the given strikethrough value.
isSet The value used to set the strikethrough flag.

Overload 2

```
public:ITextStrikethroughStyle( const ITextStrikethroughStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextStrikethroughStyle object that is copied from the given one.
other The ITextStrikethroughStyle object that is copied.

Overload 3

```
public:ITextStrikethroughStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Assignment Operator

Use this operator to assign the given ITextStrikethroughStyle object to the target one.

operator =

```
public:ITextStrikethroughStyle&
  operator =( const ITextStrikethroughStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.
other The ITextStrikethroughStyle object to be assigned.

Comparing ITextStrikethroughStyle Objects

Use these operators to compare the given ITextStrikethroughStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const ITextStrikethroughStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextStrikethroughStyle object with the current one for inequality.
other The ITextStrikethroughStyle object to be compared.

operator ==

```
public:bool
  operator ==( const ITextStrikethroughStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextStrikethroughStyle object with the current one for equality.
other The ITextStrikethroughStyle object to be compared.

Getting and Setting the Strikethrough Flag

Use these member functions to get the current setting of the strikethrough flag or to set it.

isStrikethrough

```
public:bool isStrikethrough() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current setting of the strikethrough flag. TRUE means strikethrough will be applied to the text.

setStrikethrough

```
public:void setStrikethrough(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the ITextStrikethroughStyle object's strikethrough flag using the given value.

value The value used to set the strikethrough flag.

ITextStrikethroughStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextStrikethroughStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet isContainedInSet removeFromSet	629 629 629	getValueFromSet isValueContainedInSet	629 629

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextStyle

The ITextStyle class defines the protocol shared by all text-style classes.

Do not derive your own classes from this class.

Class header file: itxtstyl.hpp

ITextStyle - Hierarchy List

ITextStyle
- ICharacterStyle
- IParagraphStyle

ITextStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextStyle objects.

~ITextStyle

```
public:virtual ~ITextStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextStyle object.

ITextStyle

Constructs an ITextStyle object.

Overload 1

```
protected:ITextStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an empty ITextStyle object.

Overload 2

```
protected:ITextStyle(const ITextStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextStyleObject that is copied from the given one.

ITextStyle

A constant reference to the ITextStyle object that is to be copied.

Assignment Operator

Use this operator to assign one ITextStyle object to another.

operator =

```
protected:ITextStyle& operator =(const ITextStyle&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Determining if the Text Style Is Contained in the Current One

Use these member functions to test whether the current ITextStyle object contains the given ITextStyleSet or value.

isContainedInSet

```
protected:virtual bool
  isContainedInSet( const ITextStyleSet& set ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests to determine if the given ITextStyleSet is contained in this, the current, set.

isValueContainedInSet

```
protected:virtual bool
  isValueContainedInSet( const ITextStyleSet& set ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests to determine if the value specified by the given ITextStyleSet is contained in this, the current, set.

Getting, Setting, and Removing ITextStyle Values

Use these member functions to get one or more style values from the set, add them to it, or remove them.

addToSet

```
protected:virtual void addToSet(ITextStyleSet& set) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the given text style set to the current one.

getValueFromSet

```
protected:virtual void getValueFromSet(const ITextStyleSet& set) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obtains the text style from the given set.

removeFromSet

```
protected:virtual void removeFromSet(ITextStyleSet& set) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the given text style from the current one.

Obtaining the Propagation Type

These member functions to get the style's propagation.

propagates

```
public:virtual EStylePropagation propagates() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the style's propagation type. This is a pure virtual function that must be overridden.

ITextStyle - Enumerations

EStylePropagation

Use the constants defined by this enumeration to specify the type of style to propagate:

kPropagateByCharacter
Styles with this propagation type can be applied to any arbitrary range of text.

kPropagateByParagraph
Styles with this propagation type can be applied only to whole paragraphs. The addStyles() and removeStyles() calls expand their ranges to whole paragraphs when used to apply or remove paragraph styles. If a character operation results in a paragraph with more than one set of paragraph styles in effect over its range, the "last" set of styles (the ones in effect at the highest character offset in the paragraph) have their ranges extended to cover the whole paragraph.

kMaximumPropagation
This value is used in some IText functions that retrieve style information to tell the function to match all style propagation types.

ITextStyle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextStyleRunIterator

The ITextStyleRunIterator class is a convenience class for walking through the style runs in an IText (i.e., the ranges over which the styles are consistent). You can specify a propagation type, which will iterate across the ranges over which the styles of the specified propagation type are consistent (regardless of what the other styles are doing). An unstyled IText is considered to have a single style run with an empty style set.

Do not derive your own classes from this class.

Class header file: istylset.hpp

ITextStyleRunIterator - Hierarchy List

ITextStyleRunIterator

ITextStyleRunIterator - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextStyleRunIterator objects.

~ITextStyleRunIterator

```
public:~ITextStyleRunIterator()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITextStyleRunIterator object.

ITextStyleRunIterator

Constructs an ITextStyleRunIterator object.

Overload 1

```
public:ITextStyleRunIterator(const ITextStyleRunIterator& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextStyleRunIterator that is copied from the given one.

Overload 2

```
public:ITextStyleRunIterator( const IText& theIText,
   ITextStyle::EStylePropagation propagation = ITextStyle::kMaximumPropagation )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Initializes the iterator to refer to the first style run in theIText.

theIText
the IText whose style runs we want to examine

propagation
the propagation type we want to match. Only styles of the specified propagation type are considered in determining run lengths and positions. kMaximumPropagation causes the iterator to consider styles of all propagation types.

Accessing Style Runs

Use these member functions to iterate over and get information about style runs.

end

```
public:static const ITextStyleRunIterator end()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

An iterator can be compared against the one returned by this function to test for loop termination. Returns an invalid iterator.

operator *

```
public:const ITextStyleSet& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the style set belonging to the current style run.

operator ++

Advances the iterator to the next style run in the IText. Returns a copy of *this from before it was advanced.

Overload 1

```
public:ITextStyleRunIterator operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITextStyleRunIterator& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator --

Advances the iterator to the previous style run in the IText. Returs a copy of *this from before it was advanced.

Overload 1

```
public:ITextStyleRunIterator& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITextStyleRunIterator operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ->

```
public:const ITextStyleSet* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows direct access to the members of the style set belonging to the curent style run.

operator bool

```
public:operator bool() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The iterator goes invalid when it is advanced off of either end of the IText's style run array. This operator can be used to test for loop termination. Returns true when the iterator is no longer valid.

runLength

```
public:length_type runLength() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of characters in the current style run.

runStart

```
public:offset_type runStart() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character offset of the first character in the current style run.

setCharacterOffset

```
public:void setCharacterOffset(offset_type charOffset)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the iterator to point to the style run containing the given character offset.

Exception

Exception	Description
IInvalidParameter	if charOffset is greater than the length of the IText being examined

setPropagation

```
public:void
  setPropagation( ITextStyle::EStylePropagation propagation )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Because there is no way to map style-run positions and lengths from one propagation type to another, this also resets the iterator to point to the first style run in the IText (according to the new propagation type).

setText

```
public:void setText(const IText& theText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the iterator to point to the first style run in the specified IText.

Comparing Style Runs

Use these member functions to compare style run iterators.

operator !=

```
public:bool operator !=(const ITextStyleRunIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given iterator is unequal to the current iterator.

operator ==

```
public:bool operator ==(const ITextStyleRunIterator& that) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given iterator is equal to the current iterator.

ITextStyleRunIterator - Type Definitions

length_type

```
typedef ioc::isize_t length_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

offset_type

```
typedef ioc::isize_t offset_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITextStyleRunIterator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITextStyleSet

The ITextStyleSet class represents a collection of text styles. It can contain any number of styles, including zero, but only one of each particular class.

Do not derive your own classes from this class.

Class header file: istylset.hpp

ITextStyleSet - Hierarchy List

IMRefCounted

ITextStyleSet

ITextStyleSet - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextStyleSet objects.

~ITextStyleSet

```
public:~ITextStyleSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextStyleSet

Constructs an ITextStyleSet object.

Overload 1

```
public:ITextStyleSet(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a style set containing the only given style.

style The style to add to the newly created style set.

Overload 2

```
public:ITextStyleSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an empty style set.

Overload 3

```
public:ITextStyleSet(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a copy of the given style set.

other The set of styles to add to the newly created style set.

Adding and Removing Contents

Use these member functions to add, remove, intersect, filter, and assign text styles.

add

Add the given style or styles to this.

Overload 1

```
public:void add(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there are any styles in other that are of the same class as styles in this, the styles in this take on the values of the styles in other

other The set of styles to add.

Overload 2

```
public:void add(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is already a style of the same class as style in this, it takes on the value of style.

style The style to add.

filterByType

```
public:void filterByType(ITextStyle::EStylePropagation typeToKeep)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Remove from this all styles that aren't of the specified propagation type.

typeToKeep
The type to keep. kMaximumPropagation is legal, but has no effect.

intersect

```
public:void intersect(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Remove from this any style that doesn't match the class of a style in other.

other The set of styles to intersect.

intersectByValue

```
public:void intersectByValue(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Remove from this any style that doesn't match both the class AND VALUE of a style in other.

other The set of styles to intersect.

operator +

Return a style set containing the styles in this plus the given style or styles.

Overload 1

```
public:ITextStyleSet operator +(const ITextStyle& style) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this to achieve the same effect as ITextStyleSet(this) += style.

style The style to add.

Overload 2

```
public:ITextStyleSet operator +(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this to achieve the same effect as ITextStyleSet(this) += other.

other The set of styles to add.

operator +=

Add the given style or styles to this. This has the same effect as add().

Overload 1

```
public:ITextStyleSet& operator +=(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is already a style of the same class as style in this, it takes on the value of style.

style The style to add.

Overload 2

```
public:ITextStyleSet& operator +=(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there are any styles in other that are of the same class as styles in this, the styles in this take on the values of the styles in other.

other The set of styles to add.

operator -

Return a style set containing the styles in this minus the given style or styles.

Overload 1

```
public:ITextStyleSet operator -(const ITextStyle& style) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this to achieve the same effect as ITextStyleSet(this) -= style.

style The style to subtract.

Overload 2

```
public:ITextStyleSet operator -(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this to achieve the same effect as ITextStyleSet(this) -= other.

other The set of styles to subtract.

operator -=

Remove the given style or styles from this. This has the same effect as remove().

Overload 1

```
public:ITextStyleSet& operator -=(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is a style in this that has the same class as style (value doesn't count, only class), it is removed.

style The style to remove.

Overload 2

```
public:ITextStyleSet& operator -=(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is any style in this that has the same class as a style in other (value doesn't count, only class), it is removed.

other The set of styles to remove.

operator =

Replace the styles in this with the given style or styles.

Overload 1

```
public:ITextStyleSet& operator =(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replace the styles in this with the styles in other.

other The set of styles to assign.

Overload 2

```
public:ITextStyleSet& operator =(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replace the styles in this with the given style.
style The style to assign.

remove

Remove from this any style that matches the class of the given style or styles.

Overload 1

```
public:void remove(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

For every style in other, if there is a style of the same class in this, it is removed. (Values don't have to match; only classes.)

other The set of styles to remove.

Overload 2

```
public:void remove(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is a style in this that has the same class as style (value doesn't count, only class), it is removed.

style The style to remove.

removeAll

```
public:void removeAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Remove all styles from this.

removeByValue

Remove from this any style that matches both the class AND VALUE of the given style or styles.

Overload 1

```
public:void removeByValue(const ITextStyle& style)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If there is a style in this that matches style in BOTH CLASS AND VALUE, it is removed.

style The style to remove.

Overload 2

```
public:void removeByValue(const ITextStyleSet& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

For every style in other, if there is a style in this that matches it in BOTH CLASS AND VALUE, it is removed.

other The set of styles to remove.

Comparing

Use these operators to compare styles sets.

operator !=

```
public:bool operator !=(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return true if the given style set and this do not contain the same styles with the same values.

other The style set to compare.

operator ==

```
public:bool operator ==(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return true if the given style set and this contain the same styles with the same values.

other The style set to compare.

Creating Empty Style Sets

Use this member function to create empty style sets.

emptyStyleSet

```
public:static ITextStyleSet& emptyStyleSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an empty style set.

Debugging

Use these methods to debug styles sets.

printDebugInfo

```
public:void printDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the classes and values of all the styles in the set, in readable form, to the location specified by class ITrace.

Extracting Contents

Use these member functions to extract the contents of styles sets.

extract

Extracts a style or styles.

Overload 1

```
public:bool
  extract( ITextStyleSet& other,
           ITextStyle::EStylePropagation propagation ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts all styles of the given propagation type.

other The style set whose contents will be replaced by all styles in this with the given propagation type.

propagation
The propagation type. kMaximumPropagation can be used to extract all styles (this would be the same as using the copy constructor).

The return value is true if the set contains any styles of the given propagation type.

Overload 2

```
public:bool extract(ITextStyle& style) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a style value.

style The style to extract. If this contains a style of the same class as style, the value of style is filled in with the value in the style set. Otherwise, style remains unchanged.

The return value is true if the set contains a style of the same class as style.

Obtaining Content Information

Use these member functions to obtain information about the contents of style sets.

contains

Returns true if the style set contains the specified style or styles.

Overload 1

```
public:bool contains(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests for style containment on a whole group of styles at once.

other The set of styles to locate.

The return value is true if for each of the styles in other there exists a style in this of the same class (value doesn't count).

Overload 2

```
public:bool contains(const ITextStyle& style) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the set contains a style of the same class as style (value doesn't count).

style The style to locate.

Overload 3

```
public:bool
  contains( ITextStyle::EStylePropagation propagation ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the style set contains any styles with the specified propagation type.

propagation
The propagation type. If kMaximumPropagation is used to mean ″all styles″ and the set contains any styles at all, the function returns true.

containsByValue

Returns true if the style set contains the specified style or styles.

Overload 1

```
public:bool containsByValue(const ITextStyleSet& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests for style containment on a whole group of styles at once.

other The set of styles to locate.

The return value is true if for each of the styles in other, this contains a style that matches it in both class AND VALUE.

Overload 2

```
public:bool containsByValue(const ITextStyle& style) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the style set contains a style which matches style in both class AND VALUE.

style The style to locate.

empty

```
public:bool empty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the set is empty (i.e., contains no styles).

styleCount

```
public:ioc::isize_t styleCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of styles contained in the set.

Streaming

Use these member functions to stream styles sets.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replace the contents of the style set with the style set data read from fromWhere.

fromWhere
The input stream.

operator >>=

```
public:void operator >>=(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes the contents of the style set to toWhere.

toWhere
The output stream.

ITextStyleSet - Inherited Member Functions and Data

Inherited Public Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
addRef removeRef	57 57	count	58

Inherited Public Data

None

Inherited Protected Functions

IMRefCounted			
Definition	**Page Number**	**Definition**	**Page Number**
~IMRefCounted operator =	57 57	IMRefCounted	57

Inherited Protected Data

None

ITextSuperSubscriptStyle

The ITextSuperSubscriptStyle class is a style that controls the baseline shift of the text it is applied to. The shift distance is expressed in points (expressed as a float), and positive values indicate upward shifts (superscripts).

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextSuperSubscriptStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextSuperSubscriptStyle

ITextSuperSubscriptStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextSuperSubscriptStyle objects.

~ITextSuperSubscriptStyle

```
public:virtual ~ITextSuperSubscriptStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ITextSuperSubscriptStyle

Constructs an ITextSuperSubscriptStyle object.

Overload 1

```
public:ITextSuperSubscriptStyle( const ITextSuperSubscriptStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextSuperSubscriptStyle object that is copied from the given one.

other The ITextSuperSubscriptStyle object that is copied.

Overload 2

```
public:ITextSuperSubscriptStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 3

```
public:ITextSuperSubscriptStyle(float offset)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextStrikethroughStyle object that is initialized to the given offset value.

offset The value used to set the offset, or baseline shift of the text.

Assignment Operator

Use this operator to assign the given ITextSuperSubscriptStyle object to the target one

operator =

```
public:ITextSuperSubscriptStyle&
  operator =( const ITextSuperSubscriptStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The ITextSuperSubscriptStyle object to be assigned.

Comparing ITextSuperSubscriptStyle Objects

Use these operators to compare the given ITextSuperSubscriptStyle object with the current one for equality or inequality.

operator !=

```
public:bool
  operator !=( const ITextSuperSubscriptStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextSuperSubscriptStyle object against the current one for inequality.

other The ITextSuperSubscriptStyle object to be tested against this one.

operator ==

```
public:bool
  operator ==( const ITextSuperSubscriptStyle& other ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextSuperSubscriptStyle object against the current one for equality.

other The ITextSuperSubscriptStyle object to be tested against this one.

Getting and Setting the Super-Subscript Offset Style Value

Use these member functions to get and set the offset value for the ITextSuperSubscriptStyle object.

offset

```
public:float offset() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current offset value setting for the ITextSuperSubscriptStyle object.

setOffset

```
public:void setOffset(float value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the offset value for the ITextSuperSubscriptStyle object.
value The value to be used to set the ITextSuperSubscriptStyle offset.

ITextSuperSubscriptStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextSuperSubscriptStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	646	getValueFromSet	646
isContainedInSet	646	isValueContainedInSet	646
removeFromSet	646		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextTypefaceStyle

The ITextTypefaceStyle class is a style that controls the typeface of a range of text. The typeface is specified by name using an IText.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextTypefaceStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextTypefaceStyle

ITextTypefaceStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextTypefaceStyle objects.

~ITextTypefaceStyle

```
public:virtual ~ITextTypefaceStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the ITextTypefaceStyle object.

ITextTypefaceStyle

Constructs an ITextTypefaceStyle object.

Overload 1

```
public:ITextTypefaceStyle(const ITextTypefaceStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextTypefaceStyle object that is copied from the given one.

other The ITextTypefaceStyle object to be copied.

Overload 2

```
public:ITextTypefaceStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 3

```
public:ITextTypefaceStyle(const IText& name)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextTypefaceStyle object that is initialized to the given typeface name.

name The typeface style to be used in constructing the ITextTypefaceStyle object.

Assignment Operator

Use this operator to assign the given ITextTypefaceStyle object to the current one.

operator =

```
public:ITextTypefaceStyle&
  operator =( const ITextTypefaceStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The ITextTypefaceStyle object to assign.

Comparing ITextTypefaceStyle Objects

Use these operators to compare the given ITextTypefaceStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextTypefaceStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextTypefaceStyle object against the current one for inequality.

other The ITextTypefaceStyle object to be compared with this one.

operator ==

```
public:bool operator ==(const ITextTypefaceStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextTypefaceStyle object against the current one for equality.

other The ITextTypefaceStyle object to be compared with this one.

Getting and Setting the Typeface Value

Use these member functions to get and set the typeface style for the ITextTypefaceStyle object.

setTypeface

```
public:void setTypeface(const IText& value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the typeface style contained in the ITextTypefaceStyle object using the given value.

value The name of the typeface.

typeface

```
public:IText typeface() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the typeface style setting of the current ITextTypefaceStyle object.

ITextTypefaceStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextTypefaceStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	649	getValueFromSet	649
isContainedInSet	649	isValueContainedInSet	649
removeFromSet	649		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextUnderlineStyle

The ITextUnderlineStyle class is a style that determines whether the text it is applied to is underlined or not. TRUE means underlined.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextUnderlineStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextUnderlineStyle

ITextUnderlineStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextUnderlineStyle objects.

~ITextUnderlineStyle

```
public:virtual ~ITextUnderlineStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITextUnderlineStyle object.

ITextUnderlineStyle

Constructs an ITextUnderlineStyle object.

Overload 1

```
public:ITextUnderlineStyle(const ITextUnderlineStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextUnderlineStyle object that is copied from the given one.
other The ITextUnderlineStyle object to be copied.

Overload 2

```
public:ITextUnderlineStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Overload 3

```
public:ITextUnderlineStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextUnderlineStyle object that is initialized to the given underline value.
isSet The value used to set the underline flag.

Assignment Operator

Use this operator to assign the given ITextUnderlineStyle object to the target one.

operator =

```
public:ITextUnderlineStyle&
  operator =( const ITextUnderlineStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The ITextUnderlineStyle object to assign.

Comparing ITextUnderlineStyle Objects

Use these operators to compare the given ITextUnderlineStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextUnderlineStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextUnderlineStyle object against the current one for inequality.

other The ITextUnderlineStyle object to be compared with this one.

operator ==

```
public:bool operator ==(const ITextUnderlineStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextUnderlineStyle object against the current one for equality.

other The ITextUnderlineStyle object to be compared with this one.

Getting and Setting the Underline Style

Use these member functions to get and set the underline style.

isUnderlined

```
public:bool isUnderlined() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current setting of the underline flag. TRUE means underline will be applied to the text.

setUnderlined

```
public:void setUnderlined(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the ITextUnderlineStyle object's underline flag using the given value.

value The value used to set the underline flag.

ITextUnderlineStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextUnderlineStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	652	getValueFromSet	652
isContainedInSet	652	isValueContainedInSet	652
removeFromSet	652		

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITextUneditableStyle

The ITextUneditableStyle class is a style that determines whether the text it is applied to is editable or not. TRUE means NOT editable.

Do not derive your own classes from this class.

Class header file: istyles.hpp

ITextUneditableStyle - Hierarchy List

ITextStyle

ICharacterStyle

ITextUneditableStyle

ITextUneditableStyle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITextUneditableStyle objects.

~ITextUneditableStyle

```
public:virtual ~ITextUneditableStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITextUneditableStyle object.

ITextUneditableStyle

Constructs an ITextUneditableStyle object.

Overload 1

```
public:ITextUneditableStyle(bool isSet)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextUneditableStyle object that is initialized to the given uneditable value.
isSet The value used to set the uneditable flag.

Overload 2

```
public:ITextUneditableStyle(const ITextUneditableStyle& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITextUneditableStyle object that is copied from the given one.
other The ITextUneditableStyle object to be copied.

Overload 3

```
public:ITextUneditableStyle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use the default constructor to create an object that leaves the style in an undefined state. You must fill in the value for a default-constructed style before you use it.

Assignment Operator

Use this operator to replace the current ITextUneditableStyle with the given one.

operator =

```
public:ITextUneditableStyle&
  operator =( const ITextUneditableStyle& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

other The ITextUneditableStyle object to assign.

Comparing ITextUneditableStyle Objects

Use these operators to compare the given ITextUneditableStyle object with the current one for equality or inequality.

operator !=

```
public:bool operator !=(const ITextUneditableStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextUneditableStyle object against the current one for inequality.

other The ITextUneditableStyle object to be compared with this one.

operator ==

```
public:bool operator ==(const ITextUneditableStyle& other) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given ITextUneditableStyle object against the current one for equality.

other The ITextUneditableStyle object to be compared with this one.

Getting and Setting the ITextUneditableStyle Value

Use these member functions to determine if ITextUneditableStyle is set, to set it to uneditable, and to get its current setting, testing if it is set to uneditable.

isUneditable

```
public:bool isUneditable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests the value of object's style to determine if it is set to uneditable.

setUneditable

```
public:void setUneditable(bool value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the style to specify that the text is not editable.

ITextUneditableStyle - Inherited Member Functions and Data

Inherited Public Functions

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ICharacterStyle	379	propagates	380

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
~ITextStyle	632	propagates	634

Inherited Public Data

None

Inherited Protected Functions

ITextStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	633	getValueFromSet	633
isContainedInSet	633	isValueContainedInSet	633
ITextStyle	632	operator =	633
removeFromSet	633		

ITextUneditableStyle			
Definition	**Page Number**	**Definition**	**Page Number**
addToSet	655	getValueFromSet	655
isContainedInSet	655	isValueContainedInSet	655
removeFromSet	655		

ICharacterStyle			
Definition	**Page Number**	**Definition**	**Page Number**
ICharacterStyle	380	operator =	380

Inherited Protected Data

None

ITime

The ITime class represents units of time (hours, minutes, and seconds) as portions of days and provides support for converting these units of time into numeric and ASCII format. You can compare and operate on ITime objects by adding them to and subtracting them from other ITime objects. The accuracy of ITime is one second in the current implementation.

A related class whose objects also represent units of time is the class IDate. The ITime class returns locale-sensitive information, based on the current locale defined at run time. See the description of the standard C function setlocale in your specific platform's system documentation for information about setting the locale.

Note: The value of ITime may go beyond the limit of a day during some operations, like addition or subtraction, and may not handle that correctly. It is recommended to use ITimeStamp for these operations.

Class header file: itime.hpp

ITime - Hierarchy List

ITime

ITime - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITime objects.

ITime

You can construct objects of this class in the following ways:

- Use the default constructor, which returns the current time.
- Give the number of seconds since midnight that the time represents. In this case, the number of seconds can be negative and is subtracted from the number of seconds in a day.
- Give the number of hours, minutes, and seconds since midnight that the time represents. In this case, the number of seconds cannot be negative.
- Copy another ITime object.
- Give a container details ICnrTime structure.

Overload 1

```
public:ITime(const ICnrTime& cnrTime)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to construct an ITime object from a container details ICnrTime structure.

cnrTime
The ICnrTime container time structure to initialize from.

Overload 2

```
public:ITime( unsigned hours,
       unsigned minutes,
       unsigned seconds = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITime object by specifying the number of hours, minutes, and seconds since midnight that the time represents. The number of hours, minutes, and seconds cannot be negative.

Overload 3

```
public:ITime(ioc::numeric_t seconds)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create an ITime object by specifying the number seconds since midnight that the time represents. If the

number of seconds is negative, it is subtracted from the number of seconds in a day and it represents a time before midnight.

```
ITime Time1(33556),     // 09:19:16
                        // 33556 seconds
                        //   = 9 hours   (32400 seconds),
                        //      19 minutes (1140 seconds),
                        //      16 seconds

ITime Time2(-33556),    // 14:40:44
                        // (9 hours, 19 minutes and
                        // 16 seconds before midnight)
```

Overload 4

```
public:ITime(const ITime& aTime)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to copy another ITime object.

Overload 5

```
public:ITime()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this default constructor to return the current time.

Comparisons

Use these member functions to compare two ITime objects.

operator !=

```
public:bool operator !=(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given time object with the current time object for inequality. Returns true if this object is not equal to the passed object, otherwise returns false.

operator <

```
public:bool operator <(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the give time object with the current time object for relative magnitude. Returns true if this object represents a time prior to the passed time.

operator <=

```
public:bool operator <=(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the give time object with the current time object for relative magnitude or equality. Returns true if this object represents a time prior to or equal to the passed time.

operator ==

```
public:bool operator ==(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given time object with the current time object for equality. Returns true if this object is equal to the passed object, otherwise returns false.

operator >

```
public:bool operator >(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given time object with the current time object for relative magnitude. Returns true if this object represents a time subsequent to the passed time.

operator >=

```
public:bool operator >=(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the given time object with the current time object for relative magnitude or equality. Returns true if this object represents a time subsequent to or equal to the passed time.

Current Time

Use this member function when you need the current time.

now

```
public:static ITime now()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current time. You can use this function as an ITime constructor.

Diagnostics

Use these member functions to provide an IString or IText representation for an ITime object. The formatting is based on the strftime conversion specifications. Often, you use these members to write trace information when debugging your code.

asString

```
public:IString
  asString( const char* fmt = IOCMSG ( ioc::kPERCENT_X ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ITime object as a string that is formatted according to the specified format string. This format string, fmt, controls how the time is formatted and can contain time ″conversion specifiers″ as defined for the standard C Library function strftime in the time.h header file. The default format is %X, which yields the time as hh:mm:ss.

The conversion specifiers that apply to ITime and their meanings are listed in the following table. IDate::asString describes conversion specifiers that apply to dates.

Specifier	Meaning
%c	Insert date and time of locale.
%H	Insert hour (24-hour clock) as a decimal number (00-23).

Specifier	Meaning
%I	Insert hour (12-hour clock) as a decimal number (01-12).
%M	Insert minute (00-59).
%p	Insert equivalent of either AM or PM locale.
%S	Insert second (00-61).
%X	Insert time representation of locale.
%Z	Insert name of time zone, or no characters if time zone is not available.
%%	Insert %.

asText

Returns the ITime object as an IText that is formatted according to the specified format string.

Overload 1

```
public:IText
  asText( const ioc::unichar_t* fmt = ( const ioc::unichar_t * )
  ( WIOCMSG ( ioc::kWPERCENT_X ) ) ) const
```

This is supported on **WIN OS/2 AIX OS/400**

Overload 2

```
public:IText
  asText( const ioc::unichar_t* fmt = ( const ioc::unichar_t * )
  ( wIocMsg_a [ ioc::kWPERCENT_X ] ) ) const
```

This is supported on **z/OS**

Manipulation

Use these member functions to update an ITime object by adding or subtracting another ITime object.

operator +

```
public:ITime operator +(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds two time objects and returns a new ITime object with the resulting time.

aTime The time to add to this time to create a new time.

operator +=

```
public:ITime& operator +=(const ITime& aTime)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds a time object to this time object. This object is modifed to reflect the new time.

aTime The time to add to this time to create a new time.

Returns a reference to this object.

operator -

```
public:ITime operator -(const ITime& aTime) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subtracts a specified time from a time object and returns a new ITime object with the resulting time.

aTime The time to subtract from this time to create a new time.

operator -=

```
public:ITime& operator -=(const ITime& aTime)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subtracts a time object from this time object. This object is modified to reflect the new time.

aTime The time to subtract from this time to create a new time.

Returns a reference to this object.

Streaming Support

Use these member functions to stream time objects. Binary streams support in and output streaming, while text streams support only output streaming.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this object in from the passed data stream. Any existing contents of this object are overwritten by the information streamed in.

fromWhere
The stream to stream in from.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams this time object out to the passed data stream.

toWhere
The stream to stream out to.

Time Queries

Use these member functions to access the seconds, minutes, and hours of an ITime object.

asICnrTime

```
public:ICnrTime asICnrTime() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a container ICnrTime structure which represents this object's time information and returns it by value.

asSeconds

```
public:ioc::numeric_t asSeconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of seconds since midnight represented by this time object.

hours

```
public:unsigned hours() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of hours past midnight represented by this object.

minutes

```
public:unsigned minutes() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of minutes past the hour represented by this object.

seconds

```
public:unsigned seconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of seconds past the minute represented by this object.

ITime - Associated Globals

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const ITime& aTime )
```

Streams this object out to an output stream, formatted as text.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const ITime& aTime )
```

Streams this object out to an output stream, formatted as text.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITime - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

ITime			
Definition	**Page Number**	**Definition**	**Page Number**
`initialize` `writeToStream`	658 658	`readFromStream`	658

Inherited Protected Data

None

ITranscoder

The ITranscoder class is the main class that provide character encodings conversion to and from Unicode.

ITranscoder provides character encodings conversion to and from Unicode. This is a platform-independent abstract class which defines the high-level bi-directional conversion protocols. It also contains static Transcoder object factory functions, exception character handling functions, query functions, and two protected pure virtual functions that mandate the implementation to be provided in the derived classes such as IPosixTranscoder, IISO8859_1Transcoder, UTF8Transcoder, etc. The ITranscoder::createTranscoder transcoder creation function takes the name of a supported character set.

The standard character code sets provided by the system or the compiler are supported. The Open Class Library also provides a number of portable code set names. The following table shows the mapping of some common code sets. The code sets listed here may not exactly match each other. Usually, code sets are located in the directory specified by the **LOCPATH** environment.

Languages	Open Class Portable Code Sets	PC Code Sets	ISO Code Sets	Microsoft LCIDs	EBCDIC-based Code Sets	EBCDIC CCSID
U.S. English, Portuguese, Canadian French	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-037	37
Danish, Norwegian	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-277	277
Finnish, Swedish	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-278	278
Italian	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-280	280
Japanese	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-281	290
Spanish	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-284	284
U.K. English	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-285	285
German	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-273	273
French	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-297	297
Belgian, Swiss German	ISO-8859-1	IBM-850	ISO8859-1	1252	IBM-500	500

Languages	Open Class Portable Code Sets	PC Code Sets	ISO Code Sets	Microsoft LCIDs	EBCDIC-based Code Sets	EBCDIC CCSID
Croatian, Czechoslo-vakian, Hungarian, Polish, Romanian, Serbian Latin, Slovak, Slovene	ISO-8859-2	IBM-852	ISO8859-2	1250	IBM-870	870
Bulgarian, Macedonian, Serbian Cyrillic, Russian	ISO-8859-5	IBM-855	ISO8859-5	1251	IBM-880 IBM-1025	880
Hebrew	ISO-8859-8	IBM-856 IBM-862	ISO8859-8	1255	IBM-424 IBM-803	424
Turkish	ISO-8859-9	IBM-857	ISO8859-9	1254	IBM-1026	1026
Arabic	ISO-8859-6	IBM-864 IBM-1046	ISO8859-6	1256	IBM-420	420
Greek	ISO-8859-7	IBM-869 (supersedes IBM-851)	ISO8859-7	1253	IBM-875	875
Japanese	EUC	IBM-943	IBM-eucJP	N/A	IBM-930, IBM-939	930
Japanese (MS compatible)	Shift-JIS	IBM-932	N/A	932	IBM-930, IBM-939	930
Korean	KSC-5601	IBM-934	IBM-eucKR	949	IBM-933	933
Traditional Chinese	CNS-11643.1986	IBM-938, big-5	IBM-eucTW	950	IBM-937	937
Simplified Chinese	GB-2312	IBM-1381	IBM-eucCN	936	IBM-935	935

NOTE: Other code sets, such as UTF-8 and the rest of the ISO8859 family are also supported. You cannot create a transcoder based on the Microsoft LCIDs. They are provided here for reference only. EBCDIC-based transcoders, however, can be created by specifying the correct code sets in the ITranscoder::createTranscoder() method.

The Transcoder Framework provides two kinds of interfaces: a high-level, easy-to-use API and a pointer-based API. The high-level API converts between a simple text string and a Unicode string by taking only two string parameters IString and IText. The pointer-based API provides a low-level conversion functionality, such that it is flexible enough to recover from conversion failure when error occurs during transcoding.

A subclass of ITranscoder must be implemented by overriding two protected pure virtual functions, doToUnicode and doFromUnicode, and other subclass-specific query functions.

Class header file: itrancod.hpp

ITranscoder - Hierarchy List

ITranscoder

ITranscoder - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ITranscoder objects.

~ITranscoder

```
public:virtual ~ITranscoder()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an ITranscoder object.

ITranscoder

Constructs an ITranscoder object.

Overload 1

```
protected:ITranscoder(const ITranscoder& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor for ITranscoder.

source The ITranscoder object copied from.

Overload 2

```
protected:ITranscoder()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Assignment Operator

Use this operator to replace the current ITranscoder object with the given one.

operator =

```
protected:ITranscoder& operator =(const ITranscoder& right)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

right The right-hand side ITranscoder to be assigned to the current one.

Character Encoding Members

Use these member functions to allow subclass providers to set the character encoding for the Transcoder.

setCharacterEncoding

```
protected:virtual void setCharacterEncoding(const IText& encoding)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows subclass providers to set the character encoding for this Transcoder.

encoding

The given character encoding.

Subclassing

If a subclass overwrites this method, be sure to call ITranscoder::setCharacterEncoding() in the overwritten method.

Conversion Members

Use the low-level conversion functions in this group to convert a Unicode text string to another encoding or a text string in another encoding to Unicode.

fromUnicode

Converts from a Unicode string to a foreign code set string.

This high-level (non-pointer-based) conversion function implicitly resets the internal Transcoding state whenever it is being called.

Overload 1

```
public:virtual result
  fromUnicode( const ioc::unichar_t* from,
               const ioc::unichar_t* from_end,
               const ioc::unichar_t *& from_next,
               char* to,
               char* to_limit,
               char *& to_next )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual result fromUnicode(const IText& from, IString& to)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

from The given Unicode text.

to The output text in foreign code set.

Returns the status of the conversion.

Overload 3

```
public:virtual result
  fromUnicode( const IText& from,
               char* to,
               char* to_limit,
               char *& to_next )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

from The given Unicode text.

to Beginning of the output conversion buffer.

to_limit

One char past the end of the output conversion buffer.

to_next

Pointer pointing one beyond the last foreign code set character successfully converted.

Returns the status of the conversion.

Overload 4

```
public:virtual result
  fromUnicode( const ioc::unichar_t* from,
               const ioc::unichar_t* from_end,
               const ioc::unichar_t *& from_next,
               IString& to )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

toUnicode

Converts from a foreign code set string to a Unicode string.

Converts from a foreign code set string to a Unicode string. Characters are translated in the range [from, from_end), placing the results in sequential position starting at ″to.″ It converts no more than (from_end - from) characters, and stores no more than (to_limit - to) characters. If it encounters a character it cannot convert, the ″exception character″ is handled according to the ″Unmapped Behavior″ for this Transcoder. For instance, the conversion stops if it encounters a character it cannot convert, and the unmapped behavior was set to kStop. It always leaves the from_next and to_next pointers pointing one beyond the last character successfully converted.

Overload 1

```
public:virtual result
  toUnicode( const char* from,
             const char* from_end,
             const char *& from_next,
             ioc::unichar_t* to,
             ioc::unichar_t* to_limit,
             ioc::unichar_t *& to_next )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual result
  toUnicode( const char* from,
             const char* from_end,
             const char *& from_next,
             IText& to )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

from Beginning of the given foreign code set text.

from_end
One char past the end of the given foreign code set text.

from_next
Pointer pointing one beyond the last foreign code set character successfully converted.

to The output Unicode text.

Returns the status of the conversion.

Overload 3

```
public:virtual result toUnicode(const IString& from, IText& to)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

from The given non-Unicode text.

to The output Unicode text.

Returns the status of the conversion.

Overload 4

```
public:virtual result
  toUnicode( const IString& from,
             ioc::unichar_t* to,
             ioc::unichar_t* to_limit,
             ioc::unichar_t *& to_next )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creating Transcoder Objects

Use these member functions to create Transcoder objects from the given values, for example, from the given character set name or from the current host character set and the kSupersetMapping proximity.

createTranscoder

Creates a transcoder.

Overload 1

```
public:static ITranscoder*
  createTranscoder( const ICCSId& id,
                    EMappingProximity proximity = kSupersetMapping )
```

This is supported on **AIX OS/400 z/OS**

Creates a Transcoder object based on the given name of the foreign character set (non-Unicode), and mapping proximity.

id The character code set identifier.

proximity
The mapping proximity

Returns a Transcoder object.

Note: The ICCSId parameter will be ignored if the open runtime routine does not support this parameter.

Overload 2

```
public:static ITranscoder* createTranscoder()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a default transcoder. Returns a Transcoder object.

Overload 3

```
public:static ITranscoder*
  createTranscoder( const IText& charSet,
                    EMappingProximity proximity = kSupersetMapping )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a Transcoder object based on the given name of the foreign character set (non-Unicode), and mapping proximity.

charSet
The character code set. This can be either the system character code set or the portable character code set.

proximity
The mapping proximity

Returns a Transcoder object.

Exception

Exception	Description
kNoAdequateTranscoder	if the given foreign character set name and the mapping proximity specified is not supported.

Exception	Description
kTranscoderNotInstalled	if the code page or conversion table for the given foreign character set is supported but not installed in the current host.

Managing the Transcoder State

Use these member functions to explicitly reset the transcoding state to be in host character set and to flush the output conversion buffer.

flush

```
public:virtual result
  flush( const char* to,
         const char* to_limit,
         char *& to_next )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Flushes the output conversion buffer so that the state of the transcoder can be made in sync with the state of ″KanjiOut,″ i.e., the ASCII state.

Note: This is only needed when clients use low-level pointer-based APIs, and want to make sure that the output conversion buffer is in ASCII state. It returns codecvt_base::partial if the buffer passed in with ″to″ and ″to_limit″ is invalid. For instance, when the current state is not in the host character set state, and (to_limit-to) < 3. The length for ″KanjiIn″ escape sequence is 3.

to beginning of the output conversion buffer.

to_limit
end of the output conversion buffer.

to_next
pointer pointing one beyond the last character successfully processed in the output conversion buffer.

Returns the status of the flush.

This function does nothing in the ITranscoder base class.

resetState

```
public:virtual void resetState()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Explicitly resets the transcoding state to be in host character set. A deriving class containing a conversion state must clear its internal transcoding state when converting a new string text.

This function does nothing in the ITranscoder base class.

Query Functions

Use these member functions to query the transcoder for information such as the character encoding it handles, the character set for the locale, the maximum number of bytes a Unicode character or character belonging to the other character set could generate, and the actual amount of storage in bytes required for the Unicode or other character.

byteBufferSize

Gets the actual amount of storage required in bytes. This information can be used to prepare the storage required for the output foreign text string that is to be converted from a given Unicode string.

Returns the actual amount of storage required to hold the text coverted from the given Unicode string.

Overload 1

```
public:virtual length_type
  byteBufferSize( const IText& uniText ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

uniText
The given Unicode string.

Overload 2

```
public:virtual length_type
  byteBufferSize( const ioc::unichar_t* from,
                  const ioc::unichar_t* from_end ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

characterEncoding

```
public:virtual IText characterEncoding() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the character encoding which this transcoder handles. For example, if the character encoding that this transcoder handles is UTF-8, then the transcoder is used to convert UTF-8 to and from Unicode.

Returns the character encoding which this transcoder handles.

characterSet

```
public:static const IText characterSet(const ILocaleKey& key)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the character set for the given locale. This character set returned is either the portable character code set (if one can be found) or the system character code set. It can then be used in ITranscoder::createTranscoder() to create a transcoder object or to create an IText object with a particular character encoding.

key The key for the given locale.

Returns the character set for the given locale.

maximumBytesPerUniChar

```
public:virtual length_type maximumBytesPerUniChar() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the maximum number of bytes generated by a Unicode character. This information can be used to prepare the storage required for strings converted from Unicode.

Returns the maximum number of bytes that can be generated by a single Unicode character.

Subclassing
This information depends on the code set conversion support by

different platforms. For example, on Win32 this is determined by querying CPINFO. For UTF8, this is set to 4.

maximumUniCharsPerByte

```
public:virtual length_type maximumUniCharsPerByte() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the maximum number of Unicode characters generated by a char-based character. This information can be used to prepare the storage required for strings converted from char-based foreign code set.

Returns the maximum number of Unicode characters that can be generated by a single char-based character.

uniCharBufferSize

Gets the actual amount of storage required in ioc::unichar_t. This information can be used to prepare the storage required for the output Unicode text string that is to be converted from a given foreign code set string.

Returns the actual amount of storage required to hold the Unicode text to be coverted from the given foreign code set string.

Overload 1

```
public:virtual length_type
  uniCharBufferSize( const char* from,
                     const char* from_end ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

from Beginning of the given foreign code set string.

from_end
End of the given foreign code set string.

Overload 2

```
public:virtual length_type
  uniCharBufferSize( const IString& text ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

text The given foreign code set string.

Substitute Character Members

Use these member functions to get the substitute character to be used for a character that cannot be directly mapped or to set the substitute character to be used.

charSubstitute

```
public:virtual char charSubstitute()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current character-based substitute character.

setCharSubstitute

```
public:virtual void setCharSubstitute(char substitute)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the character-based substitute character with the given character.

substitute
The given char-based substitute character.

uniCharSubstitute

```
public:virtual ioc::unichar_t uniCharSubstitute()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current Unicode substitute character for this transcoder.

Transcoder Subclass Conversion

Use these member functions to convert text from another character encoding to Unicode or from Unicode to another character encoding.

doFromUnicode

```
protected:virtual result
  doFromUnicode( const ioc::unichar_t* from,
                 const ioc::unichar_t* from_end,
                 const ioc::unichar_t *& from_next,
                 char* to,
                 char* to_limit,
                 char *& to_next ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts from a Unicode string to a foreign code set string.

from Beginning of the given Unicode text.

from_end
One UCS2Char past the end of the given Unicode text.

from_next
Pointer pointing one beyond the last Unicode character successfully converted.

to Beginning of the output conversion buffer.

to_limit
One char past the end of the output conversion buffer.

to_next
Pointer pointing one beyond the last foreign code set character successfully converted.

Returns the status of the conversion.

doToUnicode

```
protected:virtual result
  doToUnicode( const char* from,
               const char* from_end,
               const char *& from_next,
               ioc::unichar_t* to,
               ioc::unichar_t* to_limit,
               ioc::unichar_t *& to_next ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts from a foreign code set string to a Unicode string. This protected pure virtual function must be implemented by the derived classes which converts text string from char* to ioc::unichar_t*. Characters are translated in the range [from, from_end), placing the results in sequential position starting at ″to.″ It converts no more than (from_end - from) characters, and stores no more than (to_limit - to) characters. If it encounters a character it cannot convert, the ″exception character″ is handled according to the ″Unmapped Behavior″ for this Transcoder. For instance, the conversion stops if it encounters a character it cannot convert, and the unmapped behavior was set to kStop. It always leaves the from_next and to_next pointers pointing one beyond the last character successfully converted.

from Beginning of the given foreign code set text.

from_end
One char past the end of the given foreign code set text.

from_next
Pointer pointing one beyond the last foreign code set character successfully converted.

to Beginning of the output conversion buffer.

to_limit
One UCS2Char past the end of the output conversion buffer.

to_next
Pointer pointing one beyond the last Unicode character successfully converted.

Returns the status of the conversion.

Unmapped Character Members

Use these member functions to get and set the character handling behavior for unmapped characters.

setUnmappedBehavior

```
public:virtual void
  setUnmappedBehavior( EUnmappedBehavior unmappedBehavior )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the current unmapped character handling behavior for this transcoder.

unmappedBehavior
The given unmapped character handling behavior. Default unmapped behavior uses kUseSub.

unmappedBehavior

```
public:virtual EUnmappedBehavior unmappedBehavior()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Gets the current unmapped character handling behavior for this transcoder. Default unmapped behavior uses kUseSub before setting it using setUnmappedBehavior() function.

ITranscoder - Enumerations

EMappingProximity

The mapping proximity between character sets.

Useful constants specifying how close a transcoder being created is to the character set specified.

```
kExactMapping - Create Transcoder that exactly matches with CharSet specified.
kSupersetMapping - Create Transcoder that is a superset of CharSet specified.
kCloseMapping - Create Transcoder that is close to CharSet specified.
```

EUnmappedBehavior

Useful constants specifying how to handle unmapped characters.

These constants specify how to handle characters that cannot be converted or mapped, namely "exception characters." Exception characters are those characters whose mappings into or out of Unicode are not one-to-one.

```
kUseSub - Use substitution character if encounters exception characters.
kStop - Stop conversion if encounters exception characters.
kOmit - Omit exception characters during conversion.
```

ITranscoder - Type Definitions

length_type

```
typedef ioc::isize_t length_type
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Type used for specifying character counts.

Notes:

- Always typedef-ed to size_t because the ANSI standard uses size_t.
- Always represents character counts.

result

Type used for indicating different character code conversion results.

Variation 1

```
typedef codecvt_base::result result
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef std::codecvt_base::result result
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITranscoder - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IUnicode

The IUnicode class provides the property information for a Unicode character. The Unicode character information, provided implicitly by the Unicode character encoding standard, includes information about the script (for example, symbols or control characters) to which the character belongs, as well as semantic information such as whether a character is a digit or uppercased, lowercased, or uncased. The current implementation is based on the Unicode Standard 3.0beta.

Do not derive your own classes from this class.

Class header file: iunicode.hpp

IUnicode - Hierarchy List

IUnicode

IUnicode - Member Functions and Data by Group

Character Property Members

Use these member functions to get the Unicode character's script, character property, and direction property information.

characterDirection

```
public:static EDirectionProperty
  characterDirection( ioc::unichar_t uc )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the linguistic direction property of a character.

For example, 0x0041 (letter A) has the kLeftToRight directional property.

script

```
public:static EUnicodeScript script(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the script property of a Unicode character.

For example, 0x03A9 (Omega) is of the Greek script.

type

```
public:static ECharacterProperty type(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character type property of a Unicode character.

For example, 0x0030 (Zero) has kDecimalNumber (Numeric) type.

Identifying Special Characters

Use these member functions to identify special characters. For example, you can test a character to determine if it is a graphic character, a control character, a punctuation mark, and so on.

digitValue

```
public:static int digitValue(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds the numeric value of the Unicode character which represents a decimal digit. For example, '0', '1', '2', ...; returns 0, 1, 2. This function is used so that non-Roman decimal digits can also be used in numbers, such as the Arabic numerals at U+0660 to U+0669.

uc The Unicode character

isASCII

```
public:static bool isASCII(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an ASCII character.

uc The Unicode character

isClosePunctuation

```
public:static bool isClosePunctuation(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an closing punctuation. For example, left parenthesis, or closing single quotation mark.
uc The Unicode character

isControl

```
public:static bool isControl(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a control character.
uc The Unicode character

isDigit

```
public:static bool isDigit(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is numeric.
uc The Unicode character

isGraphic

```
public:static bool isGraphic(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is graphical. For example, special control characters.
uc The Unicode character

isHexDigit

```
public:static bool isHexDigit(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a hex digit.
uc The Unicode character

isInSet

```
public:static bool isInSet(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a valid Unicode character.
uc The Unicode character

isOpenPunctuation

```
public:static bool isOpenPunctuation(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an opening punctuation. For example, right parenthesis, or opening single quotation mark.
uc The Unicode character

isPrint

```
public:static bool isPrint(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is printable.
uc The Unicode character

isPunctuation

```
public:static bool isPunctuation(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a punctuation.
uc The Unicode character

isSymbol

```
public:static bool isSymbol(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a symbol. For example, the math symbols, the division sign.
uc The Unicode character

isXDigit

```
public:static bool isXDigit(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a hex digit.
uc The Unicode character

matchPunctuation

```
public:static ioc::unichar_t
  matchPunctuation( ioc::unichar_t searchChar )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Finds the matching punctuation for the Unicode character. For example, left parenthesis for the right parenthesis, or vice versa.
uc The Unicode character

Obtaining Character Information

Use these member functions to test whether a character is a line separator, a paragraph separator, a space, an invisible character, or a trailing invisible character.

isASpace

```
public:static bool isASpace(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a space character.
uc The Unicode character

isInvisible

```
public:static bool isInvisible(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an invisible character. For example, white space or line terminator.
uc The Unicode character

isLineOrParagraphSeparator

```
public:static bool isLineOrParagraphSeparator(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a line or paragraph separator.
uc The Unicode character

isLineSeparator

```
public:static bool isLineSeparator(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a line separator.
uc The Unicode character

isParagraphSeparator

```
public:static bool isParagraphSeparator(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a paragraph separator.
uc The Unicode character

isTrailingInvisible

```
public:static bool isTrailingInvisible(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a trailing invisible character. For example, non-break white spaces.
uc The Unicode character

Testing for Base Form Characters and Diacritics

Use these member functions to determine if a character is a base form letter, a number, or a diacritic, and, if it is a letter, whether it is uppercase, lowercase, or uncased.

isAlpha

```
public:static bool isAlpha(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is alphabetic.
uc The Unicode character

isAlphaNumeric

```
public:static bool isAlphaNumeric(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an alphabetic letter or a number.
uc The Unicode character

isBaseForm

```
public:static bool isBaseForm(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a based form letter.
uc The Unicode character

isDiacritic

```
public:static bool isDiacritic(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is a diacritic mark.
uc The Unicode character

isLower

```
public:static bool isLower(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is lowercase.
uc The Unicode character

isUncased

```
public:static bool isUncased(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is an uncased letter.
uc The Unicode character

isUpper

```
public:static bool isUpper(ioc::unichar_t uc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Checks if a character is uppercase.
uc The Unicode character

Unicode Version Members

Use this member function to identify the version of the Unicode standard used for this implementation.

currentVersion

```
public:static double currentVersion()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the version of the Unicode standard this implementation is based on.

IUnicode - Enumerations

ECharacterProperty

Enumerates character properties.

EDirectionProperty

Enumerates values that specify the language directional property of a character set.

EUnicodeBounds

EUnicodeBounds enumerates the following constants:

- kLowBoundUnicode 0x0000
- kLowBoundASCII kLowBoundUnicode
- kLowBoundLatinOne kLowBoundASCII
- kHighBoundASCII 0x007F
- kHighBoundLatinOne 0x00FF
- kLowBoundHan 0x4E00
- kHighBoundHan 0x9FA5
- kHighBoundHangulSyllable 0xD7A3 Specifies the upper limit of currently defined precomposed Hangul syllable range.

- kLowBoundUserZone 0xE000 Specifies the beginning of general user characters.
- kHighBoundUserZone 0xF8FF Specifies the beginning of corporate user characters.
- kLowBoundDefinedUserZone kHighBoundUserZone - 255, Specifes the bottom end of defined corporate user characters.
- kLowBoundCompatibilityZone1 kHighBoundUserZone + 1 CJK compatibility, Arabic forms, and so forth.
- kHighBoundCompatibilityZone1 0xFEFE
- kLowBoundCompatibilityZone2 0xFF00 Half width characters
- kHighBoundCompatibilityZone2 0xFFEF
- kHighBoundUnicode 0xFFFF

EUnicodeScript

Enumerates all the scripts implemented in Unicode 2.0.

IUnicode - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UAPLFunctionalSymbol

The APL functional symbols.

Class header file: iusyms.hpp

UAPLFunctionalSymbol - Hierarchy List

UAPLFunctionalSymbol

UAPLFunctionalSymbol - Member Functions and Data by Group

UAPLFunctionalSymbol - Enumerations

EAPLFunctionalSymbolNames

Unicode 2.0 APL functional symbol names.

UAPLFunctionalSymbol - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArabic

The Arabic characters.

Class header file: iumeast.hpp

UArabic - Hierarchy List

UArabic

UArabic - Member Functions and Data by Group

UArabic - Enumerations

EArabicNames

Unicode 2.0 Arabic character names.

UArabic - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArabicCompatibility

The Arabic presentation forms.

Class header file: iucmpbty.hpp

UArabicCompatibility - Hierarchy List

UArabicCompatibility

UArabicCompatibility - Member Functions and Data by Group

UArabicCompatibility - Enumerations

EArabicCompatibilityNames

Unicode 2.0 Arabic compatibility character names.

UArabicCompatibility - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArabicLigature

The Arabic ligatures.

Class header file: iucmpbty.hpp

UArabicLigature - Hierarchy List

UArabicLigature

UArabicLigature - Member Functions and Data by Group

UArabicLigature - Enumerations

EArabicLigatureNames

Unicode 2.0 Arabic ligature names.

UArabicLigature - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArmenian

The Armenian characters.

Class header file: iumeast.hpp

UArmenian - Hierarchy List

UArmenian

UArmenian - Member Functions and Data by Group

UArmenian - Enumerations

EArmenianNames

UArmenian - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArmenianSmallLigature

The American small ligatures.

Class header file: iucmpbty.hpp

UArmenianSmallLigature - Hierarchy List

UArmenianSmallLigature

UArmenianSmallLigature - Member Functions and Data by Group

UArmenianSmallLigature - Enumerations

EArmenianSmallLigatureNames

Unicode 2.0 Armenian small ligature names.

UArmenianSmallLigature - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UArrow

The arrow characters.

Class header file: iusyms.hpp

UArrow - Hierarchy List

UArrow

UArrow - Member Functions and Data by Group

UArrow - Enumerations

EArrowNames

Unicode 2.0 arrow character names.

UArrow - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UASCII

The ASCII characters.

Class header file: iugnrl.hpp

UASCII - Hierarchy List

UASCII

UASCII - Member Functions and Data by Group

UASCII - Enumerations

EASCIINames

UASCII - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UBengali

The Bengali signs and letters.

Class header file: iusasia.hpp

UBengali - Hierarchy List

UBengali

UBengali - Member Functions and Data by Group

UBengali - Enumerations

EBengaliNames

Unicode 2.0 Bengali character names.

UBengali - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UBlocks

The block elements.

Class header file: iucmpbty.hpp

UBlocks - Hierarchy List

UBlocks

UBlocks - Member Functions and Data by Group

UBlocks - Enumerations

EBlocksNames

UBlocks - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UBopomofo

The Chinese pronounciation sound marks. Based on GB 2312.

Class header file: iueasia.hpp

UBopomofo - Hierarchy List

UBopomofo

UBopomofo - Member Functions and Data by Group

UBopomofo - Enumerations

EBopomofoNames

Unicode 2.0 Bopomofo letter names.

UBopomofo - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UBoxDrawings

The box drawing characters , for the form and chart components.

Class header file: iucmpbty.hpp

UBoxDrawings - Hierarchy List

UBoxDrawings

UBoxDrawings - Member Functions and Data by Group

UBoxDrawings - Enumerations

EBoxDrawingsNames

Unicode 2.0 box drawing character names.

UBoxDrawings - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledDigit

The circled digits.

Class header file: iucmpbty.hpp

UCircledDigit - Hierarchy List

UCircledDigit

UCircledDigit - Member Functions and Data by Group

UCircledDigit - Enumerations

ECircledDigitNames

Unicode 2.0 circled digit names.

UCircledDigit - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledHangul

The circled Hangul elements.

Class header file: iucmpbty.hpp

UCircledHangul - Hierarchy List

UCircledHangul

UCircledHangul - Member Functions and Data by Group

UCircledHangul - Enumerations

ECircledHangulNames

Unicode 2.0 circled Hangul names.

UCircledHangul - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledIdeograph

The circled ideographs.

Class header file: iucmpbty.hpp

UCircledIdeograph - Hierarchy List

UCircledIdeograph

UCircledIdeograph - Member Functions and Data by Group

UCircledIdeograph - Enumerations

ECircledIdeographNames

Unicode 2.0 circled ideograph names.

UCircledIdeograph - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledKatakana

The circled Katakana.

Class header file: iucmpbty.hpp

UCircledKatakana - Hierarchy List

UCircledKatakana

UCircledKatakana - Member Functions and Data by Group

UCircledKatakana - Enumerations

ECircledKatakanaNames

Unicode 2.0 circled katakana names.

UCircledKatakana - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledLatin

The circled Latin letters.

Class header file: iucmpbty.hpp

UCircledLatin - Hierarchy List

UCircledLatin

UCircledLatin - Member Functions and Data by Group

UCircledLatin - Enumerations

ECircledLatinNames

Unicode 2.0 circled Latin character names.

UCircledLatin - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCircledNumber

The circled numbers.

Class header file: iucmpbty.hpp

UCircledNumber - Hierarchy List

UCircledNumber

UCircledNumber - Member Functions and Data by Group

UCircledNumber - Enumerations

ECircledNumberNames

Unicode 2.0 circled number names.

UCircledNumber - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCJKCompatibility

The CJK compatibility ideographs.

Class header file: iucmpbty.hpp

UCJKCompatibility - Hierarchy List

UCJKCompatibility

UCJKCompatibility - Member Functions and Data by Group

UCJKCompatibility - Enumerations

ECJKCompatibilityNames

Unicode 2.0 CJK compatiblity character names.

UCJKCompatibility - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCJKRadical

CJK radicals

Class header file: iueasia.hpp

UCJKRadical - Hierarchy List

UCJKRadical

UCJKRadical - Member Functions and Data by Group

UCJKRadical - Enumerations

CJKRadicalNames

UCJKRadical - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCJKSquaredAbbreviations

The squared Katakana words and Latin abbreviations.

Class header file: iucmpbty.hpp

UCJKSquaredAbbreviations - Hierarchy List

UCJKSquaredAbbreviations

UCJKSquaredAbbreviations - Member Functions and Data by Group

UCJKSquaredAbbreviations - Enumerations

ECJKSquaredAbbreviationsNames

Unicode 2.0 squared Japanese word and Latin abbreviation names.

UCJKSquaredAbbreviations - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCJKSquaredWords

The squared Katakana words, Japanese era names, and Japanese corporation.

Class header file: iucmpbty.hpp

UCJKSquaredWords - Hierarchy List

UCJKSquaredWords

UCJKSquaredWords - Member Functions and Data by Group

UCJKSquaredWords - Enumerations

ECJKSquaredWordsNames

Unicode 2.0 sqared CJK names.

UCJKSquaredWords - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCJKSymbols

The CJK symbols, punctuations, numerals and diacritcs.

Class header file: iusyms.hpp

UCJKSymbols - Hierarchy List

UCJKSymbols

UCJKSymbols - Member Functions and Data by Group

UCJKSymbols - Enumerations

ECJKSymbolsNames

UCJKSymbols - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCNSCompatibility

The CNS compatibility characters.

Class header file: iucmpbty.hpp

UCNSCompatibility - Hierarchy List

UCNSCompatibility

UCNSCompatibility - Member Functions and Data by Group

UCNSCompatibility - Enumerations

ECNSCompatibilityNames

UCNSCompatibility - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCombining

The combining characters which includes the ordinary diacritics.

Class header file: iugnrl.hpp

UCombining - Hierarchy List

UCombining

UCombining - Member Functions and Data by Group

UCombining - Enumerations

ECombiningNames

Unicode 2.0 combining character names.

UCombining - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UControlCode

The control characters.

Class header file: iugnrl.hpp

UControlCode - Hierarchy List

UControlCode

UControlCode - Member Functions and Data by Group

UControlCode - Enumerations

EControlCodeNames

Unicode 2.0 control character names.

UControlCode - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UControlCodePicture

The graphic pictures for control codes.

Class header file: iusyms.hpp

UControlCodePicture - Hierarchy List

UControlCodePicture

UControlCodePicture - Member Functions and Data by Group

UControlCodePicture - Enumerations

EControlCodePictureNames

Unicode 2.0 control code graphic picture names.

UControlCodePicture - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCoptic

Coptic unique letters.

Class header file: iueeuro.hpp

UCoptic - Hierarchy List

UCoptic

UCoptic - Member Functions and Data by Group

UCoptic - Enumerations

ECopticNames

Unicode 2.0 Coptic-unique letter names.

UCoptic - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCurrency

The currency signs, including kEuroSign (0x20AC).

Class header file: iusyms.hpp

UCurrency - Hierarchy List

UCurrency

UCurrency - Member Functions and Data by Group

UCurrency - Enumerations

ECurrencyNames

Unicode 2.0 currency sign names, including kEuroSign (0x20AC). kEuroCurrencySign (0x20A0) is obsolete and should not be used. If needed, you may use kHistoricEuroSign for the Unicode character 0x20A0.

UCurrency - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UCyrillic

Cyrillic signs and letters.

Class header file: iueeuro.hpp

UCyrillic - Hierarchy List

UCyrillic

UCyrillic - Member Functions and Data by Group

UCyrillic - Enumerations

ECyrillicNames

Unicode 2.0 Cyrillic character names.

UCyrillic - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UDevanagari

The Devanagari signs and letters.

Class header file: iusasia.hpp

UDevanagari - Hierarchy List

UDevanagari

UDevanagari - Member Functions and Data by Group

UDevanagari - Enumerations

EDevanagariNames

Unicode 2.0 Devanagari character names.

UDevanagari - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UDingbats

The miscellaneous symbols, dingbats.

Class header file: iusyms.hpp

UDingbats - Hierarchy List

UDingbats

UDingbats - Member Functions and Data by Group

UDingbats - Enumerations

EDingbatsNames

UDingbats - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UFullStopDigit

The digits with a period.

Class header file: iucmpbty.hpp

UFullStopDigit - Hierarchy List

UFullStopDigit

UFullStopDigit - Member Functions and Data by Group

UFullStopDigit - Enumerations

EFullStopDigitNames

Unicode 2.0 fullstop digit names.

UFullStopDigit - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UFullStopNumber

The numbers with a period.

Class header file: iucmpbty.hpp

UFullStopNumber - Hierarchy List

UFullStopNumber

UFullStopNumber - Member Functions and Data by Group

UFullStopNumber - Enumerations

EFullStopNumberNames

Unicode 2.0 fullstop number names.

UFullStopNumber - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UFullwidth

The fullwidth variants.

Class header file: iucmpbty.hpp

UFullwidth - Hierarchy List

UFullwidth

UFullwidth - Member Functions and Data by Group

UFullwidth - Enumerations

EFullwidthNames

Unicode 2.0 fullwidth character names.

UFullwidth - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGeneralPunctuation

The general punctuation characters.

Class header file: iugnrl.hpp

UGeneralPunctuation - Hierarchy List

UGeneralPunctuation

UGeneralPunctuation - Member Functions and Data by Group

UGeneralPunctuation - Enumerations

EGeneralPunctuationNames

UGeneralPunctuation - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGeometricShapes

The geometric shapes.

Class header file: iusyms.hpp

UGeometricShapes - Hierarchy List

UGeometricShapes

UGeometricShapes - Member Functions and Data by Group

UGeometricShapes - Enumerations

EGeometricShapesNames

Unicode 2.0 geometric shape names.

UGeometricShapes - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGeorgian

Georgian signs and letters.

Class header file: iueeuro.hpp

UGeorgian - Hierarchy List

UGeorgian

UGeorgian - Member Functions and Data by Group

UGeorgian - Enumerations

EGeorgianNames

Unicode 2.0 Georgian character names.

UGeorgian - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGreek

Greek diaciritcs, signs and letters.

Class header file: iueeuro.hpp

UGreek - Hierarchy List

UGreek

UGreek - Member Functions and Data by Group

UGreek - Enumerations

EGreekNames

UGreek - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGujarati

The Gujarati signs and letters.

Class header file: iusasia.hpp

UGujarati - Hierarchy List

UGujarati

UGujarati - Member Functions and Data by Group

UGujarati - Enumerations

EGujaratiNames

Unicode 2.0 Gujarati character names.

UGujarati - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UGurmukhi

The Gurmukhi signs and letters.

Class header file: iusasia.hpp

UGurmukhi - Hierarchy List

UGurmukhi

UGurmukhi - Member Functions and Data by Group

UGurmukhi - Enumerations

EGurmukhiNames

Unicode 2.0 Gurmukhi character names.

UGurmukhi - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHalfwidth

The halfwidth variants.

Class header file: iucmpbty.hpp

UHalfwidth - Hierarchy List

UHalfwidth

UHalfwidth - Member Functions and Data by Group

UHalfwidth - Enumerations

EHalfwidthNames

Unicode 2.0 halfwidth character names.

UHalfwidth - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHalfwidthHangulLetter

The halfwidth Hangul variants.

Class header file: iucmpbty.hpp

UHalfwidthHangulLetter - Hierarchy List

UHalfwidthHangulLetter

UHalfwidthHangulLetter - Member Functions and Data by Group

UHalfwidthHangulLetter - Enumerations

EHalfwidthHangulLetterNames

Unicode 2.0 halfwidth Hangul names.

UHalfwidthHangulLetter - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangul

Class header file: iueasia.hpp

UHangul - Hierarchy List

UHangul

UHangul - Member Functions and Data by Group

UHangul - Enumerations

EHangulNames

UHangul - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangulChoseong

The Hangul Cheoseong. The Korean initial consonants.

Class header file: iueasia.hpp

UHangulChoseong - Hierarchy List

UHangulChoseong

UHangulChoseong - Member Functions and Data by Group

UHangulChoseong - Enumerations

EHangulChoseongNames

Unicode 2.0 Hangul Cheoseong names.

UHangulChoseong - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangulJongseong

The Hangul Jongseong. Korean final consonants.

Class header file: iueasia.hpp

UHangulJongseong - Hierarchy List

UHangulJongseong

UHangulJongseong - Member Functions and Data by Group

UHangulJongseong - Enumerations

EHangulJongseongNames

Unicode 2.0 Hangul Jongseong names.

UHangulJongseong - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangulJungseong

The Hangul Jungseong. The Korean medial vowels.

Class header file: iueasia.hpp

UHangulJungseong - Hierarchy List

UHangulJungseong

UHangulJungseong - Member Functions and Data by Group

UHangulJungseong - Enumerations

EHangulJungseongNames

Unicode 2.0 Hangul Jungseong names.

UHangulJungseong - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangulLetter

The Hungul modern letters. Based on KS C 5601.

Class header file: iueasia.hpp

UHangulLetter - Hierarchy List

UHangulLetter

UHangulLetter - Member Functions and Data by Group

UHangulLetter - Enumerations

EHangulLetterNames

Unicode 2.0 Hangul letter names.

UHangulLetter - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangulSyllable

The Hangul syllables.

Class header file: iucmpbty.hpp

UHangulSyllable - Hierarchy List

UHangulSyllable

UHangulSyllable - Member Functions and Data by Group

UHangulSyllable - Enumerations

EHangulSyllableNames

Unicode 2.0 Hangul syllable names.

UHangulSyllable - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHangzhouNumeral

The Hangzhou style numerals.

Class header file: iueasia.hpp

UHangzhouNumeral - Hierarchy List

UHangzhouNumeral

UHangzhouNumeral - Member Functions and Data by Group

UHangzhouNumeral - Enumerations

EHangzhouNumeralNames

Unicode 2.0 Hangzhou numeral names.

UHangzhouNumeral - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHanNumeral

The Han numerals.

Class header file: iueasia.hpp

UHanNumeral - Hierarchy List

UHanNumeral

UHanNumeral - Member Functions and Data by Group

UHanNumeral - Enumerations

EHanNumeralNames

Unicode 2.0 Han Numeral names.

UHanNumeral - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHarpoon

The Harpoon symbols.

Class header file: iusyms.hpp

UHarpoon - Hierarchy List

UHarpoon

UHarpoon - Member Functions and Data by Group

UHarpoon - Enumerations

EHarpoonNames

Unicode 2.0 Harpoon symbol names.

UHarpoon - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHebrew

The Hebrew characters.

Class header file: iumeast.hpp

UHebrew - Hierarchy List

UHebrew

UHebrew - Member Functions and Data by Group

UHebrew - Enumerations

EHebrewNames

UHebrew - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UHiragana

The Hiragana letters. Based on JIS X 0208.

Class header file: iueasia.hpp

UHiragana - Hierarchy List

UHiragana

UHiragana - Member Functions and Data by Group

UHiragana - Enumerations

EHiraganaNames

Unicode 2.0 Hiragana letter names.

UHiragana - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UIdeographicAnnotation

The ideographic annotations. The Kanbun.

Class header file: iueasia.hpp

UIdeographicAnnotation - Hierarchy List

UIdeographicAnnotation

UIdeographicAnnotation - Member Functions and Data by Group

UIdeographicAnnotation - Enumerations

EIdeographicAnnotationNames

UIdeographicAnnotation - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UIdeographicTelegraph

The ideographic telegraph symbols.

Class header file: iucmpbty.hpp

UIdeographicTelegraph - Hierarchy List

UIdeographicTelegraph

UIdeographicTelegraph - Member Functions and Data by Group

UIdeographicTelegraph - Enumerations

EIdeographicTelegraphNames

Unicode 2.0 ideographic telegraph symbol names.

UIdeographicTelegraph - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UKangXiRadical

The KangXi radicals.

Class header file: iueasia.hpp

UKangXiRadical - Hierarchy List

UKangXiRadical

UKangXiRadical - Member Functions and Data by Group

UKangXiRadical - Enumerations

EKangXiRadicalNames

Unicode 2.0 KangXi radical names.

UKangXiRadical - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UKannada

The Kannada signs and letters.

Class header file: iusasia.hpp

UKannada - Hierarchy List

UKannada

UKannada - Member Functions and Data by Group

UKannada - Enumerations

EKannadaNames

Unicode 2.0 kannada character names.

UKannada - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UKatakana

The Katakana letters. Based on JIS X 0208.

Class header file: iueasia.hpp

UKatakana - Hierarchy List

UKatakana

UKatakana - Member Functions and Data by Group

UKatakana - Enumerations

EKatakanaNames

Unicode 2.0 Katakana letter names.

UKatakana - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UKatakanaHiragana

The Katakana/Hiragana sound marks.

Class header file: iueasia.hpp

UKatakanaHiragana - Hierarchy List

UKatakanaHiragana

UKatakanaHiragana - Member Functions and Data by Group

UKatakanaHiragana - Enumerations

EKatakanaHiraganaNames

Unicode 2.0 Katakana/Hiragana sound mark names.

UKatakanaHiragana - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ULao

The Lao signs and letters.

Class header file: iusasia.hpp

ULao - Hierarchy List

ULao

ULao - Member Functions and Data by Group

ULao - Enumerations

ELaoNames

Unicode 2.0 Lao character names.

ULao - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ULatin

The Latin characters.

Class header file: iugnrl.hpp

ULatin - Hierarchy List

ULatin

ULatin - Member Functions and Data by Group

ULatin - Enumerations

ELatinNames

Unicode 2.0 Latin character names.

ULatin - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ULatin1

The Latin-1 characters. (aka ISO 8859-1)

Class header file: iugnrl.hpp

ULatin1 - Hierarchy List

ULatin1

ULatin1 - Member Functions and Data by Group

ULatin1 - Enumerations

ELatin1Names

ULatin1 - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ULatinSmallLigature

The Latin small ligatures.

Class header file: iucmpbty.hpp

ULatinSmallLigature - Hierarchy List

ULatinSmallLigature

ULatinSmallLigature - Member Functions and Data by Group

ULatinSmallLigature - Enumerations

ELatinSmallLigatureNames

Unicode 2.0 Latin small ligature names.

ULatinSmallLigature - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ULetterLikeSymbol

The letter-like symbols.

Class header file: iusyms.hpp

ULetterLikeSymbol - Hierarchy List

ULetterLikeSymbol

ULetterLikeSymbol - Member Functions and Data by Group

ULetterLikeSymbol - Enumerations

ELetterLikeSymbolNames

ULetterLikeSymbol - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UMalayalam

The Malayalam signs and letters.

Class header file: iusasia.hpp

UMalayalam - Hierarchy List

UMalayalam

UMalayalam - Member Functions and Data by Group

UMalayalam - Enumerations

EMalayalamNames

Unicode 2.0 Malayalam character names.

UMalayalam - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UMathematicalOperators

The mathematical operator signs.

Class header file: iusyms.hpp

UMathematicalOperators - Hierarchy List

UMathematicalOperators

UMathematicalOperators - Member Functions and Data by Group

UMathematicalOperators - Enumerations

EMathematicalOperatorsNames

UMathematicalOperators - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UMiscellaneousTechnical

The miscellaneous technical signs.

Class header file: iusyms.hpp

UMiscellaneousTechnical - Hierarchy List

UMiscellaneousTechnical

UMiscellaneousTechnical - Member Functions and Data by Group

UMiscellaneousTechnical - Enumerations

EMiscellaneousTechnicalNames

UMiscellaneousTechnical - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UModifierLetter

The phonetic modifiers derived from Latin characters.

Class header file: iugnrl.hpp

UModifierLetter - Hierarchy List

UModifierLetter

UModifierLetter - Member Functions and Data by Group

UModifierLetter - Enumerations

EModifierLetterNames

Unicode 2.0 Latin modifier names.

UModifierLetter - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UOCR

The OCR symbols.

Class header file: iusyms.hpp

UOCR - Hierarchy List

UOCR

UOCR - Member Functions and Data by Group

UOCR - Enumerations

EOCRNames

Unicode 2.0 OCR symbol names.

UOCR - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UOriya

The Oriya signs and letters.

Class header file: iusasia.hpp

UOriya - Hierarchy List

UOriya

UOriya - Member Functions and Data by Group

UOriya - Enumerations

EOriyaNames

Unicode 2.0 Oriya character names.

UOriya - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UParenthesizedDigit

The parenthesized digits.

Class header file: iucmpbty.hpp

UParenthesizedDigit - Hierarchy List

UParenthesizedDigit

UParenthesizedDigit - Member Functions and Data by Group

UParenthesizedDigit - Enumerations

EParenthesizedDigitNames

Unicode 2.0 Parenthesized digit names.

UParenthesizedDigit - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UParenthesizedHangul

The parenthesized Hangul elements.

Class header file: iucmpbty.hpp

UParenthesizedHangul - Hierarchy List

UParenthesizedHangul

UParenthesizedHangul - Member Functions and Data by Group

UParenthesizedHangul - Enumerations

EParenthesizedHangulNames

Unicode 2.0 parenthesized Hangul names.

UParenthesizedHangul - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UParenthesizedIdeograph

The parenthesized ideographs.

Class header file: iucmpbty.hpp

UParenthesizedIdeograph - Hierarchy List

UParenthesizedIdeograph

UParenthesizedIdeograph - Member Functions and Data by Group

UParenthesizedIdeograph - Enumerations

EParenthesizedIdeographNames

Unicode 2.0 parenthesized ideograph names.

UParenthesizedIdeograph - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UParenthesizedLatin

The parenthesized Latin letters.

Class header file: iucmpbty.hpp

UParenthesizedLatin - Hierarchy List

UParenthesizedLatin

UParenthesizedLatin - Member Functions and Data by Group

UParenthesizedLatin - Enumerations

EParenthesizedLatinNames

Unicode 2.0 parenthesized Latin character names.

UParenthesizedLatin - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UParenthesizedNumber

The parenthesized numbers.

Class header file: iucmpbty.hpp

UParenthesizedNumber - Hierarchy List

UParenthesizedNumber

UParenthesizedNumber - Member Functions and Data by Group

UParenthesizedNumber - Enumerations

EParenthesizedNumberNames

Unicode 2.0 parenthesized number names.

UParenthesizedNumber - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UPresentationFormForVertical

The glyphs for vertical variants.

Class header file: iucmpbty.hpp

UPresentationFormForVertical - Hierarchy List

UPresentationFormForVertical

UPresentationFormForVertical - Member Functions and Data by Group

UPresentationFormForVertical - Enumerations

EPresentationFormForVerticalNames

Unicode 2.0 vertical variant glyph names.

UPresentationFormForVertical - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

URomanNumeral

The Roman numerals.

Class header file: iucmpbty.hpp

URomanNumeral - Hierarchy List

URomanNumeral

URomanNumeral - Member Functions and Data by Group

URomanNumeral - Enumerations

ERomanNumeralNames

Unicode 2.0 Roman numeral names.

URomanNumeral - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

USmallVariants

The small form variants.

Class header file: iucmpbty.hpp

USmallVariants - Hierarchy List

USmallVariants

USmallVariants - Member Functions and Data by Group

USmallVariants - Enumerations

ESmallVariantsNames

USmallVariants - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UStandardPhonetic

The phonetic symbols, IPA extensions.

Class header file: iusyms.hpp

UStandardPhonetic - Hierarchy List

UStandardPhonetic

UStandardPhonetic - Member Functions and Data by Group

UStandardPhonetic - Enumerations

EStandardPhoneticNames

Unicode 2.0 phonetic symbol names.

UStandardPhonetic - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

USubscript

The subscripts.

Class header file: iucmpbty.hpp

USubscript - Hierarchy List

USubscript

USubscript - Member Functions and Data by Group

USubscript - Enumerations

ESubscriptNames

Unicode 2.0 subscript names.

USubscript - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

USuperscript

The superscripts.

Class header file: iucmpbty.hpp

USuperscript - Hierarchy List

USuperscript

USuperscript - Member Functions and Data by Group

USuperscript - Enumerations

ESuperscriptNames

Unicode 2.0 superscript names.

USuperscript - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

USyriac

Class header file: iumeast.hpp

USyriac - Hierarchy List

USyriac

USyriac - Member Functions and Data by Group

USyriac - Enumerations

ESyriacNames

USyriac - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UTamil

The Tamil signs and letters.

Class header file: iusasia.hpp

UTamil - Hierarchy List

UTamil

UTamil - Member Functions and Data by Group

UTamil - Enumerations

ETamilNames

Unicode 2.0 Tamil character names.

UTamil - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UTelugu

The Telugu signs and letters.

Class header file: iusasia.hpp

UTelugu - Hierarchy List

UTelugu

UTelugu - Member Functions and Data by Group

UTelugu - Enumerations

ETeluguNames

Unicode 2.0 Telugu character names.

UTelugu - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UThai

The Thai signs and letters.

Class header file: iusasia.hpp

UThai - Hierarchy List

UThai

UThai - Member Functions and Data by Group

UThai - Enumerations

EThaiNames

Unicode 2.0 Thai character names.

UThai - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UUnicodeDigit

The Unicode digit characters.

Class header file: iugnrl.hpp

UUnicodeDigit - Hierarchy List

UUnicodeDigit

UUnicodeDigit - Member Functions and Data by Group

UUnicodeDigit - Enumerations

EUnicodeDigitNames

Unicode 2.0 digit character names.

UUnicodeDigit - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UUnicodeSpecial

The special characters, including invalid characters 0xFFFE and 0xFFFF.

Class header file: iugnrl.hpp

UUnicodeSpecial - Hierarchy List

UUnicodeSpecial

UUnicodeSpecial - Member Functions and Data by Group

UUnicodeSpecial - Enumerations

EUnicodeSpecialNames

Unicode 2.0 special character names.

UUnicodeSpecial - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UVulgarFraction

The vulgar fraction characters.

Class header file: iucmpbty.hpp

UVulgarFraction - Hierarchy List

UVulgarFraction

UVulgarFraction - Member Functions and Data by Group

UVulgarFraction - Enumerations

EVulgarFractionNames

Unicode 2.0 vulgar fraction number names.

UVulgarFraction - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UWhiteSpace

The white space characters.

Class header file: iugnrl.hpp

UWhiteSpace - Hierarchy List

UWhiteSpace

UWhiteSpace - Member Functions and Data by Group

UWhiteSpace - Enumerations

EWhiteSpaceNames

UWhiteSpace - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

UZapfDingbats

The Zapf dingbats.

Class header file: iusyms.hpp

UZapfDingbats - Hierarchy List

UZapfDingbats

UZapfDingbats - Member Functions and Data by Group

UZapfDingbats - Enumerations

EZapfDingbatsNames

Unicode 2.0 Zapf dingbats symbol names.

UZapfDingbats - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 5. Error Handling, Tracing, and Testing Classes

IAccessError

The IAccessError class represents an exception. When a member function makes a request of the operating system or the presentation system that the system cannot satisfy, the member function creates and throws an object of the IAccessError class. If the operating system or the presentation system cannot satisfy the request due to resource exhaustion, member functions create and throw objects of the class IResourceExhausted.

Note: Typically, if no other exception fits an error condition, the Open Class Library creates and throws an object of the IAccessError class.

Class header file: iexcbase.hpp

IAccessError - Hierarchy List

IException

IAccessError

IAlreadyExists

IEntityInUse

IMustBeEmpty

IObjectNotFound

IAccessError - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IAccessError object from another.

~IAccessError

```
public:virtual ~IAccessError()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IAccessError

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IAccessError( const char* errorText,
                     ioc::unumeric_t errorId = 0,
                     Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

© Copyright IBM Corp. 2001

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IAccessError( const ioc::unichar_t* errorText,
              ioc::unumeric_t errorId = 0,
              Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IAccessError(const IAccessError& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IAccessError - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IAlreadyExists

IAlreadyExists is an IException derived class. IAlreadyExists provides a message reporting that the function could not create the requested object because the name of the object or its location is already used. You can use instances of this class to signal that your code encounted circumstances causing it to throw the exception. If you want to provide more specific information, IAlreadyExists allows you to construct a custom message and use it instead of the default one.

Class header file: igenexc.hpp

IAlreadyExists - Hierarchy List

IException

IAccessError

IAlreadyExists

IAlreadyExists - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IAlreadyExists object from another.

~IAlreadyExists

```
public:virtual ~IAlreadyExists()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IAlreadyExists

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IAlreadyExists(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to create an exception that allows you to specify a customized error message. If you provide one, your error message is used instead of the default message when an object of the same name or location as an existing object is requested.

text The error message to use for this exception. If you do not provide an error message, a default message is used.

Overload 2

```
public:IAlreadyExists(const IAlreadyExists&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IAlreadyExists - Inherited Member Functions and Data

Inherited Public Functions

IAccessError			
Definition	**Page Number**	**Definition**	**Page Number**
~IAccessError	731	IAccessError	731
nameW	732		

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IArgumentDictionary

IArgumentDictionary is a general-purpose class for parsing text arguments on a command line and creating an ordered collection of key/value pairs. It parses arguments from an ITest object, a single string, or UNIX-style argc/argv parameters.

A leading hyphen character identifies keywords. Anything without a leading hyphen is a value argument. A keyword picks up the following argument if it is not another keyword. This example shows how IArgumentDictionary parses the command line input to create the key/value pairs.

```
Command-line input:
  -parm -n sample1 Sue Tom -ccc 84 85
```

This table lists the options defined for the example:

Key	Value	Description
-parm		The value associated with -parm is an empty IText, not NIL.
-n	sample1	The -n argument picks up the following argument.
1	Sue	There is no associated key so ″1″ is assigned.
2	Tom	There is no associated key so ″2″ is assigned.
-ccc	84	The -ccc argument picks up the value ″84″.
3	85	There is no associated key so ″3″ is assigned.

Class header file: iargdict.hpp

IArgumentDictionary - Hierarchy List

IArgumentDictionary

IArgumentDictionary - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~IArgumentDictionary

```
public:virtual ~IArgumentDictionary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor.

IArgumentDictionary

IArgumentDictionary provides several constructors with a variety of signatures. Each constructor calls a corresponding importTextArguments function. Some constructors accept an optional parameter that specifies the options to be treated as naked options. (Naked options are input options that never take values in a key/value pair.)

Overload 1

```
public:IArgumentDictionary( ITest& theTest,
  const IString& nakedOptions = IArgumentDictionary::kEmptyText )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes a constant reference to a ITest object and uses the inputs in theTest as arguments.

This unique constructor uses the outputTextStream() of the ITest object for its own error message output. When this constructor is used within an ITest, a failure to parse a number (which typically causes the test to fail) produces an error message in the diagnostic output of the ITest. If a different constructor is used, the error message appears but is lost thereafter. It can then be difficult to determine what caused the test to fail.

theTest
Test object.

nakedOptions
The options that never take values.

Overload 2

```
public:IArgumentDictionary( const IString& argumentString,
  const IString& nakedOptions = IArgumentDictionary::kEmptyText )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor that takes an argument input string as input to the constructor.

argumentString
The argument string.

nakedOptions
The options that never take values.

Overload 3

```
public:IArgumentDictionary(const IArgumentDictionary& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

source The IArgumentDictionary object to be copied.

Overload 4

```
public:IArgumentDictionary( const int argc,
   const char * const* argv,
   const IString& nakedOptions = IArgumentDictionary::kEmptyText )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor that takes standard UNIX style argc and argv arguments as inputs.

argc The standard argc argument as passed in from the command line.

argv The standard argv argument as passed in from the command line.

nakedOptions
The options that never take values.

Overload 5

```
public:IArgumentDictionary()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Assignment

Standard assignment operators.

operator =

```
public:IArgumentDictionary&
  operator =( const IArgumentDictionary& rhs )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator for the IArgumentDictionary class.

rhs The IArgumentDictionary object used to do the assignment.

Importing and Exporting

Member functions used to import and export information to and from the IArgumentDictionary.

exportTextArguments

```
public:void exportTextArguments(IString& argumentString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function exports the arguments in the dictionary as a string of space-delimited keys and values.

argumentString

The string of keys and values.

importTextArguments

Copies input text arguments from the passed-in parameters.

Places the key/value pairs from the input arguments in the dictionary. Arguments are added to any existing dictionary entries. Options are recognized by a leading '-' character. IArgumentDictionary tries to recognize negative numbers so you can not have input options that could be confused with negative numbers. For example, "-2ab" is a valid option; "-123" is not.

Overload 1

```
public:void
  importTextArguments( const int argc,
                       const char * const* argv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the UNIX style argc and argv key/value input text parameters.

argc UNIX-style argument count.

argv UNIX-style argument vector.

Overload 2

```
public:void importTextArguments(const IString& argumentString)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the white-spaced delimited string of keys and values from the input argument string.

argumentString

Name of a white-space delimited string of keys and values.

Overload 3

```
public:void importTextArguments(ITest& theTest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies text arguments from the ITest object parameter.

theTest

The ITest object where the text arguments should be copied from.

specifyNakedOptions

```
public:void specifyNakedOptions(const IString& nakedOptions)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies that the given options never take values in a key/value pair.

Call these methods **before** calling ImportTextArguments to change the way input arguments are handled.

nakedOptions

IString containing a space delineated list of the options that never takes values (e.g., (″-p -i -q″)

Lookup Functions

Functions to control the lookup of key/value pairs and to lookup values in the IArgumentDictionary based on a given key or index.

findSimilarKey

```
public:bool
  findSimilarKey( const IString& keyToMatch,
                  IString& newKey ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function looks for a key similar to the value passed in the argument keyToMatch. If a similar key is found, it is returned in the argument newKey.

Note: key is considered to match if it is of equal length or longer than the match key, and if all characters in the match key are equal (case insensitive) to those in the key. Call lookupSimilarKeys(true) to always have the dictionary look for similar keys.

keyToMatch

Name of the key to match.

newKey

Name of the key similar to keyToMatch.

Return

false if no similar key is found; otherwise true.

lookupSimilarKeys

```
public:void lookupSimilarKeys(bool)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function toggles on or off the use of the findSimilarKey lookups. If lookupSimilarKeys(true) is called, the dictionary will henceforth call findSimilarKey for all lookups. It will also match similar keys during importTextArguments calls when checking to see if keys are naked keys.

bool true/false

nthNumber

This function returns the index-th option-less value parsed as a number in the parameter value. It behaves like the range-checked numberAt methods. The number must fall in the range minAllowed and maxAllowed.

Overload 1

```
public:bool
  nthNumber( const short index,
             ioc::numeric_t& value,
             ioc::numeric_t minAllowed = LONG_MIN,
             ioc::numeric_t maxAllowed = LONG_MAX )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool
  nthNumber( const short index,
             double& value,
             double minAllowed = - DBL_MAX,
             double maxAllowed = DBL_MAX )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the index-th option-less value parsed as a number in the parameter value.

index Index number of the value that is being looked up.

value Retrieved value (double).

minAllowed

Minimum value allowed (double).

maxAllowed

Maximum value allowed (double).

Return

true if the argument exists, it parses perfectly, and it is within the given range. false if the number parses badly, is out of range, or the argument does not exist, leaving value unchanged.

nthValue

```
public:bool nthValue(const short index, IString& value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function attempts to retrieve the index-th option-less value. It returns false if no such value exists.

index The given index.

value IString for retrieved value.

Return

false if no such value exists; otherwise true.

numberAt

Looks up a number argument corresponding to the given key. The number must fall in the range minAllowed <= x <= maxAllowed. If the number parses bad, the argument does not exist, or is out of range, false is returned and ″value″ remains unchanged.

Overload 1

```
public:bool
  numberAt( const IString& key,
            ioc::numeric_t& value,
            ioc::numeric_t minAllowed = LONG_MIN,
            ioc::numeric_t maxAllowed = LONG_MAX )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool
  numberAt( const IString& key,
            double& value,
            double minAllowed = - DBL_MAX,
            double maxAllowed = DBL_MAX )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

key Key of the value that is being looked up.

value Retrieved value (double).

minAllowed
Minimum value allowed (double).

maxAllowed
Maximum value allowed (double).

Return

true if the argument exists, it parses perfectly and is within the given range, inclusively. false if the number parses badly, is out of range, or the argument does not exist, leaving value unchanged.

removeOnLookup

```
public:void removeOnLookup(bool)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Causes key/value pairs to be removed from the dictionary as they are looked up.

If removeOnLookup is called with argument true, then keys and values will be removed from the dictionary (keys will also be deleted) as they are looked up. This is useful if you want to make sure that all arguments are parsed by:

- calling removeOnLookup(true) to turn on automatic key/value pair removal
- deleting each argument as you retrieve it
- checking count() == 0 to make sure there are no leftover arguments.

bool true to turn on automatic key/value pair removal; false to turn it off

textAt

```
public:bool textAt(const IString& key, IString& value)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value associated with the given key.

If the value is numeric, textAt returns it as a string. If no value is given, an empty IString object is returned.

key The given key for lookup.

value The retrieved value, if one is found.

Return

false if no such key exists; otherwise true.

Miscellaneous

Miscellaneous functions for the IArgumentDictionary.

countOfArguments

```
public:ioc::unumeric_t countOfArguments() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function returns the number of key/value pairs that are currently stored in the dictionary.

Public Constants

Public constants.

kEmptyText

```
public:static const IString & kEmptyText
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Used to represent empty text fields in the IArgumentDictionary class.

IArgumentDictionary - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IAssertionFailure

The IAssertionFailure class represents an exception. The IASSERT macro expands to create and throw an object of the IAssertionFailure class if the specified condition is not met. An **assertion** is a debugging tool you can use to assure a condition is true. The class IException describes IASSERT and the other exception-handling macros.

Class header file: iexcbase.hpp

IAssertionFailure - Hierarchy List

IException

IAssertionFailure

IAssertionFailure - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IAssertionFailure object from another.

~IAssertionFailure

```
public:virtual ~IAssertionFailure()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IAssertionFailure

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macro IASSERT. The Open Class Library provides this macro to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IAssertionFailure( const ioc::unichar_t* errorText,
                          ioc::unumeric_t errorId = 0,
                          Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IAssertionFailure( const char* errorText,
                          ioc::unumeric_t errorId = 0,
                          Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IAssertionFailure(const IAssertionFailure& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IAssertionFailure - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IBaseErrorInfo

The IBaseErrorInfo class is an abstract base class that defines the interface for its derived classes. These classes retrieve error information and text that you can subsequently use to create an exception object. The following macros assist in throwing exceptions:

IASSERTPARM

This macro accepts an expression to test. The expression is asserted to be true. If it evaluates to false, the macro generates code that calls the IExcept__assertParameter function, which creates an IInvalidParameter exception. The error group, other, is added to the object. The exception data is logged using IException::TraceFn::logExceptionData, and the exception is then thrown.

IASSERTSTATE

This macro accepts an expression to test. The expression is asserted to be true. If it evaluates to false, the macro generates code that calls the IExcept__assertState function, which creates an IInvalidRequest exception. The error group, other, is added to the object. The exception data is logged, and the exception is then thrown.

ITHROWLIBRARYERROR

This macro can throw any of the Open Class Library-defined exceptions.

id The ID of the message to load from the class library message file.

name A value from the enumeration IBaseErrorInfo::ExceptionType, indicating the type of exception to create.

severity

A value from the enumeration IException::Severity, indicating the severity of the exception.

The macro generates code that calls the IExcept__throwLibraryError function, which does the following:

1. Loads the message text from the class library message file
2. Uses the message text to create an exception object
3. Adds location information
4. Logs the exception data
5. Throws the exception

ITHROWLIBRARYERROR1

This macro can throw any of the Open Class Library-defined exceptions. It is identical to the ITHROWLIBRARYERROR macro, except it has a fourth parameter:

text Replacement text for the retrieved message.

ITHROWERROR

This macro can throw any of the Open Class Library-defined exceptions.

messageId

The ID of the message to load from the message file.

name A value from the enumeration IBaseErrorInfo::ExceptionType, indicating the type of exception to create.

severity
A value from the enumeration IException::Severity, indicating the severity of the exception.

messageFile
The name of the message file to load the exception text from. This name should include the file extension, for example, usermsg.msg.

errorGroup
The errorGroup associated with this error. This can be one of the values for ErrorCodeGroup defined in IException or a value you provide.

The macro generates code that calls the IExcept__throwError function, which does the following:
1. Loads the message text from the specified library message file
2. Uses the message text to create an exception object
3. Adds the error group to the object
4. Adds location information
5. Logs the exception data
6. Throws the exception

ITHROWERROR1
This macro can throw any of the Open Class Library-defined exceptions. It is identical to the ITHROWERROR macro, except it has a fourth parameter:

substitutionText
Substitution text for the retrieved message.

Class header file: iexcbase.hpp

IBaseErrorInfo - Hierarchy List

IBaseErrorInfo
ICLibErrorInfo
ISystemErrorInfo

IBaseErrorInfo - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class. You cannot copy or assign objects of this class.

~IBaseErrorInfo

```
public:virtual ~IBaseErrorInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IBaseErrorInfo

```
public:IBaseErrorInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IBaseErrorInfo object.

Error Information

Use these member functions to return error information provided by objects of this class.

errorId

```
public:virtual ioc::unumeric_t errorId() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error ID.

isAvailable

```
public:virtual bool isAvailable() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If error information is available, true is returned.

text

```
public:virtual const char* text() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error text.

textW

```
public:virtual const ioc::unichar_t* textW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error text in Unicode format.

Throw Support

Use these member functions to support the throwing of exceptions using information from an IBaseErrorInfo object. The throwError function is used by the ITHROWERROR macro.

throwError

```
public:void
  throwError( const IExceptionLocation& location,
              ExceptionType name = accessError,
              IException::Severity severity = IException::recoverable,
              IException::ErrorCodeGroup errorGroup = IException::baseLibrary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an IBaseErrorInfo object and uses it to do the following:

1. Create an exception object
2. Add the error code group to the object
3. Add the location information to the object
4. Log the exception data
5. Throw the exception

location

An IExceptionLocation object containing the following:

- Function name
- File name
- Line number where the function is called

name Use the enumeration ExceptionType to specify the type of the exception. The default is accessError.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

errorGroup

Use one of the ErrorCodeGroup values provided in IException, or provide your own group for this parameter. The default is baseLibrary.

throwErrorW

```
public:void
  throwErrorW( const IExceptionLocation& location,
               ExceptionType name = accessError,
               IException::Severity severity = IException::recoverable,
               IException::ErrorCodeGroup errorGroup = IException::baseLibrary )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of throwError(). Creates an IBaseErrorInfo object and uses it to do the following:

1. Create an exception object
2. Add the error code group to the object
3. Add the location information to the object
4. Log the exception data
5. Throw the exception

location

An IExceptionLocation object containing the following:

- Function name
- File name
- Line number where the function is called

name Use the enumeration ExceptionType to specify the type of the exception. The default is accessError.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

errorGroup

Use one of the ErrorCodeGroup values provided in IException, or provide your own group for this parameter. The default is baseLibrary.

IBaseErrorInfo - Enumerations

ExceptionType

Use these enumerators to specify the type of exception to create.

OS/400 Considerations

AS/400 C++ does not support the outOfWindowResource exception type.

z/OS Considerations

OS/390® C++ does not support the outOfWindowResource exception type.

IBaseErrorInfo - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICannotProceed

The ICannotProceed exception creates an error message that you can use to report that your code encountered a situation prohibiting it from completing an operation. You should throw this exception when it may be possible to complete the operation another way or recover from the error condition.

If you want to provide a specific, detailed exception message, ICannotProceed allows you to construct one and use it instead of the default message.

Using inherited member functions, you can obtain an error ID identifying what error caused the exception and a severity code identifying whether the exception can be recovered from.

Class header file: igenexc.hpp

ICannotProceed - Hierarchy List

IException

ICannotProceed

ICannotProceed - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one ICannotProceed object from another.

~ICannotProceed

```
public:virtual ~ICannotProceed()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICannotProceed

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICannotProceed(const ICannotProceed&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:ICannotProceed(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an exception that allows you to specify a customized error message. If you provide one, your error message is used instead of the default one when an operation cannot complete, but recovery or an alternative means of completion is possible.

text The error message to use for this exception. If you do not provide an error message, a default message is used.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

ICannotProceed - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
nameW	770	setErrorCodeGroup	766
setErrorId	767	setSeverity	769
setText	769	setTextW	769
setTraceFunction	768	terminate	765
text	770	textCount	770
textW	770		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766

IException			
Definition	**Page Number**	**Definition**	**Page Number**
operatingSystem presentationSystem	766 767	other	767

Inherited Protected Functions

None

Inherited Protected Data

None

ICLibErrorInfo

The ICLibErrorInfo class represents error information. When a C library call results in an error condition, objects of the ICLibErrorInfo class are created. The per thread global variable errno is used to obtain the error text.

The Open Class Library provides the ITHROWCLIBERROR macro for throwing exceptions constructed with ICLibErrorInfo information. This macro has the following parameters:

location
The name of the C function returning the error code, the name of the file the function is in, and the function's line number.

name Use the enumeration IBaseErrorInfo::ExceptionType to specify the type of the exception. The default is accessError.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

This macro generates code that calls throwCLibError, which does the following:
1. Creates an ICLibErrorInfo object
2. Uses the object to create an IException object
3. Adds the CLibrary error group to the object
4. Adds location information
5. Logs the exception data
6. Throws the exception

If Unicode error info is required, use the ITHROWCLIBERRORW macro. The macro generates code that calls throwCLibErrorW, which logs the exception data in Unicode character string.

Class header file: iexcbase.hpp

ICLibErrorInfo - Hierarchy List

IBaseErrorInfo

ICLibErrorInfo

ICLibErrorInfo - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class. You cannot copy or assign objects of this class.

~ICLibErrorInfo

```
public:virtual ~ICLibErrorInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICLibErrorInfo

Constructs an ICLibErrorInfo object.

Overload 1

```
public:ICLibErrorInfo(const ioc::unichar_t* CLibFunctionName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ICLibErrorInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Overload 3

```
public:ICLibErrorInfo(const char* CLibFunctionName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an object of this class.

Note: If the constructor cannot load the error text, the Open Class Library provides the following default text: ″No error text is available.″

CLibFunctionName
The name of the failing C library function. If you specify **CLibFunctionName**, the constructor prefixes it to the error text. Optional.

Error Information

Use these member functions to return error information provided by objects of this class.

errorId

```
public:virtual ioc::unumeric_t errorId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error ID.

isAvailable

```
public:virtual bool isAvailable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the error text is available, true is returned.

text

```
public:virtual const char* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error text.

textW

```
public:virtual const ioc::unichar_t* textW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error text in Unicode format.

Throw Support

Use these member functions to support the throwing of exceptions using information from an ICLibErrorInfo object. The throwCLibError function is used by the ITHROWCLIBERROR macro.

throwCLibError

```
public:static void
  throwCLibError( const char* functionName,
                  const IExceptionLocation& location,
                  IBaseErrorInfo::ExceptionType name = accessError,
                  IException::Severity severity = IException::recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an ICLibErrorInfo object and uses the text from it to do the following:

1. Create an exception object
2. Add the location information to it
3. Log the exception data
4. Throw the exception

functionName
The name of the function where the exception occurred.

location
An IExceptionLocation object containing the following:
- Function name
- File name
- Line number where the function is called

name Use the enumeration IBaseErrorInfo::ExceptionType to specify the type of the exception. The default is accessError.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

throwCLibErrorW

```
public:static void
  throwCLibErrorW( const ioc::unichar_t* functionName,
                   const IExceptionLocation& location,
                   IBaseErrorInfo::ExceptionType name = accessError,
                   IException::Severity severity = IException::recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of throwCLibError(). Creates an ICLibErrorInfo object and uses the text from it to do the following:

1. Create an exception object
2. Add the location information to it
3. Log the exception data
4. Throw the exception

functionName
The name of the function where the exception occurred.

location
An IExceptionLocation object containing the following:
- Function name
- File name
- Line number where the function is called

name Use the enumeration IBaseErrorInfo::ExceptionType to specify the type of the exception. The default is accessError.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

ICLibErrorInfo - Inherited Member Functions and Data

Inherited Public Functions

IBaseErrorInfo			
Definition	**Page Number**	**Definition**	**Page Number**
~IBaseErrorInfo	746	errorId	747
IBaseErrorInfo	746	isAvailable	747
text	747	throwError	747
throwErrorW	748		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IComparisonTestOf

IComparisonTestOf tests the constructor, copy constructor and assignment operator of the target class. This is a good way to leverage testing effort since it allows you to test classes if those classes are expected to adhere to some protocol. This protocol test is implemented using a template class.

Clients can either create a subclass or directly use this class.

IComparisonTestOf tests the following operations of its target class:

```
- copy constructed stack object
- operator= assigned stack object
- operator= self-assigned stack object
- copy constructed heap object
```

```
- operator= assigned heap object
- operator= self-assigned heap object
- Separately constructed unequal object of the unequal target is given
```

Target class must have operator== defined.

The best was to use the IComparisonTest framework is to use the protocol macros ComparisonTestMacro and ComparisonTestEqualOnlyMacro.

These macros do everything you need in one line. They write the entire header and the entire implementation, including construtors and destructors. Just supply the names of the test class, target class, comparator, the parameter list for the constructor that creates the target, and (optionally) the unequal target. The parameter list may be left blank to indicate that the empty constructor is to be used to create the target or unequal target. If the test is only testing equal targets, then use the ″EqualOnly″ macros.

```
Usage:

   Example:  ISample is the target class which we want to test.  ISample
             has a constructor, a copy constructor, an operator=, and
             an operator== defined:

             comparisonTestMacro(ISampleTest, ISample, (1), (2));

   Example:  Example for a class which creates the target object using the
             empty constructor:

             comparisonTestMacro(ISampleTest, ISample, , (2));

   Now you can run ISampleTest with the following command:

             runTestImplementationMacro(ISampleTest);

   If you do not want to test your target against an unequal object of the
   same type, do the following:

             comparisonTestEqualOnlyMacro(ISampleTest, ISample, (1));

   Or you can use this global function which creates and returns an instance
   of IComparisonTestOf:

             createComparisonTest(new ISample(1), new ISample(2));

   which can be used with the adoptTest method (see the ITestCollection class).
```

Class header file: icmptest.hpp

IComparisonTestOf - Hierarchy List

ITest

IComparisonTestOf

IComparisonTestOf - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~IComparisonTestOf

```
public:virtual ~IComparisonTestOf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor

IComparisonTestOf

Use these to construct obejcts of this class.

Overload 1

```
public:IComparisonTestOf(const IComparisonTestOf < AType >& that)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Overload 2

```
public:IComparisonTestOf(AType* target, AType* unequalTarget = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor which accepts the following arguments:

target Target object to perform tests on.

unequalTarget
Unequal object to use for comparison.

Assignment Operators

Use this operator to replace the existing object with the assigned, given one.

operator =

```
public:IComparisonTestOf < AType >&
  operator =( const IComparisonTestOf < AType >& that )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator.

Test Methods

Class member functions used to by the IComparison tests. Some need to be overridden by the user and some should NOT be overridden. See each member function for details.

copyInfo

```
public:virtual void copyInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Within copyInfo, IComparisonTestOf calls the inherited implementation (e.g., IDerived::copyInfo should call IBase::copyInfo). Then copyInfo should add key/value pairs to infoMap by calling the addInfo method.

targetClassNameAsString

```
protected:virtual const char* targetClassNameAsString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the target class. By default, it returns NULL.

You must override this method if you want the target class name reported in the test output.

test

```
protected:virtual void test()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Tests the constructor, copy constructor, and assignment operator of the target class.

Note: Do NOT override the method. Do NOT call this method directly. The best way to use the IComparisonTest framework is to use the protocol macros: ComparisonTestMacro and ComparisonTestEqualOnlyMacro.

IComparisonTestOf - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInputs	811
exceptionStage	812	lookupInfo	816
operator =	812	outputTextStream	816
run	812	runCount	817
setInputs	811	success	813

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	cleanup	814
isReset	817	isSuccessUndecided	818
ITest	812	print	814
reset	814	setStickySuccess	818
setSuccess	818	setup	814
test	815		

Inherited Protected Data

None

IConditionInvalid

IConditionInvalid is an exception that indicates an attempted operation on an ICondition object which is in an invalid state.

Class header file: ithrdexc.hpp

IConditionInvalid - Hierarchy List

IException

IInvalidParameter

IConditionInvalid

IConditionInvalid - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IConditionInvalid object from another.

~IConditionInvalid

```
public:virtual ~ICondition Invalid()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IConditionInvalid

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICondition Invalid(const ICondition Invalid& toCopy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:ICondition Invalid(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the exception with the given message text.

text The error message text to use for this exception. If no text is specified, the default text is loaded from the appropriate localized message file.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IConditionInvalid - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IInvalidParameter			
Definition	**Page Number**	**Definition**	**Page Number**
~IInvalidParameter	778	IInvalidParameter	778
nameW	779		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IDeviceError

The IDeviceError class represents an exception. When a member function makes a hardware-related request of the operating system or the presentation system that the system cannot satisfy because of a hardware failure, the member function creates and throws an object of the IDeviceError class.

Class header file: iexcbase.hpp

IDeviceError - Hierarchy List

IException

IDeviceError

IVolumeOffline

IDeviceError - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IDeviceError object from another.

~IDeviceError

```
public:virtual ~IDeviceError()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IDeviceError

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IDeviceError( const char* errorText,
                     ioc::unumeric_t errorId = 0,
                     Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IDeviceError( const ioc::unichar_t* errorText,
                     ioc::unumeric_t errorId = 0,
                     Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IDeviceError(const IDeviceError& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IDeviceError - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771

IException			
Definition	**Page Number**	**Definition**	**Page Number**
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IException

The IException class is the base class from which all exception objects thrown in the library are derived. None of the functions in this class throws exceptions because an exception has probably already been thrown or is about to be thrown. Member functions in the Open Class Library create objects of classes derived from IException for all error conditions the functions encounter. Each exception object contains the following:

- A stack of exception message text strings (descriptions)
- An error ID
- A severity code
- An error code group
- Information about where the exception was thrown

IException provides all of the functions required for it and its derived classes, including functions that operate on the text strings in the stack.

The library defines the derived classes so that you can catch exceptions by their type. In general, never create an IException object. Instead, create and throw an object of the appropriate derived class.

The derived classes of IException are the following:

IAccessError

IAssertionFailure
IDeviceError
IInvalidParameter
IInvalidRequest
IResourceExhausted

In addition, IResourceExhausted has the following derived classes:

IOutOfMemory
IOutOfSystemResource
IOutOfWindowResource

Note: OS/400 and z/OS C/C++ do not support IOutOfWindowResource. It is listed here because versions of Open Class Library on other operating systems do support it.

You can also derive your own exception type from IException.

The Open Class Library provides the following macros to assist in using exception handling. If you derive your own exception type and you want to use a macro, you must use the ITHROW macro or write your own macro. Each macro has a Unicode version (in parenthesis). They are usually named by appending the letter ″W″ at the end of the macros. For example, ITHROWW is the Unicode version of ITHROW. These macros take the same arguments as their ASCII counterparts except for user supplied strings. User supplied strings should be encoded in Unicode format.

ITHROW (ITHROWW)
Accepts as input a predefined object of any IException-derived class. The macro generates code to add the location information to the objects, logs all object data, and throws the exception.

IRETHROW (IRETHROWW)
Accepts as input an object of any derived class of IException that has been previously thrown and caught. Like the ITHROW macro, it also captures the location information and logs all object data before rethrowing the exception.

IASSERT (IASSERTW)
If you define IC_DEVELOP during the compile for debugging purposes, this macro expands to provide assertion support in the library. This macro accepts an expression to test. If the test evaluation returns false, IASSERT calls assertParameter.

IEXCLASSDECLARE
Creates a declaration for a derived class of IException or one of its derived classes. Starting with Version 5 of IOC, all methods are declared ″inline″ with this macro.

IEXCLASSIMPLEMENT
Starting with Version 5 of IOC, this macro does nothing.

IEXCEPTION_LOCATION (IEXCEPTION_LOCATIONW)
Expands to create an object of the class IExceptionLocation.

INO_EXCEPTIONS_SUPPORT
Supports compilers lacking an exception-handling implementation. If you

use the INO_EXCEPTIONS_SUPPORT macro, the following macros end the program after capturing the location information and logging it. These macros normally throw an exception:

ITHROW (ITHROWW)
Found in IException.

IASSERTPARM (IASSERTPARAMW)
Found in IBaseErrorInfo.

IASSERTSTATE (IASSERTSTATEW)
Found in IBaseErrorInfo.

ITHROWERROR (ITHROWERRORW)
Found in IBaseErrorInfo.

ITHROWERROR1 (ITHROWERRORW1)
Found in IBaseErrorInfo.

ITHROWLIBRARYERROR (ITHROWLIBRARYERRORW)
Found in IBaseErrorInfo.

ITHROWLIBRARYERROR1 (ITHROWLIBRARYERRORW1)
Found in IBaseErrorInfo.

ITHROWSYSTEMERROR (ITHROWSYSTEMERRORW)
Found in ISystemErrorInfo.

Note: OS/400 and z/OS C/C++ do not support IGUIErrorInfo. It is listed here because versions of Open Class Library on other operating systems do support it.

Note: The INO_EXCEPTIONS_SUPPORT macro might not work correctly on all compilers.

Whenever the Open Class Library throws one of these exceptions, trace records are output with information about the exception. The class ITrace describes tracing in more detail.

Class header file: iexcbase.hpp

IException - Hierarchy List

IException

- IDecimalDataError
- IStreamerInvalidException
- IInvalidContextNumber
- IResourceExhausted
- IInvalidRequest
- ICollectionUnlockException
- ICollectionLockException
- IInvalidParameter
- IInvalidDataOnStream
- IWaitAbandoned
- ICannotProceed
- IAddressAlreadyInContext
- IStreamErrorException
- IDeviceError
- IAssertionFailure
- ICollectionResourceException

IAccessError
ICollectionLockTimeOutException
IPreconditionViolation
IFileOperationCancelled
IUnknownTypeOnStream

IException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IException object to another.

~IException

```
public:virtual ~IException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IException

You can construct objects of this class by doing the following:

- Using the primary constructor. Normally, this is the only way you would construct an object of this class.
- Using the copy constructor.

Overload 1

```
public:IException( const ioc::unichar_t* errorText,
            ioc::unumeric_t errorId = 0,
            Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IException( const char* errorText,
            ioc::unumeric_t errorId = 0,
            Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IException(const IException& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

The Open Class Library provides this constructor so the compiler can copy the exception when it is thrown.

exception
The exception object you want to copy.

Application Termination

These member funcitons provide support for terminating an application instead of throwing an exception.

terminate

```
public:virtual void terminate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Ends the application. Normally, the Open Class Library only intends this function to be used internally by the library's exception-handling macros when the compiler you are using does not support C++ exception handling. This only occurs if you define the INO_EXCEPTIONS_SUPPORT macro. The macros that use this function are as follows:

ITHROW
Found in IException.

IASSERTPARM
Found in IBaseErrorInfo.

IASSERTSTATE
Found in IBaseErrorInfo.

ITHROWLIBRARYERROR
Found in IBaseErrorInfo.

ITHROWLIBRARYERROR1
Found in IBaseErrorInfo

ITHROWGUIERROR
Found in IGUIErrorInfo.

ITHROWGUIERROR2
Found in IGUIErrorInfo.

ITHROWSYSTEMERROR
Found in ISystemErrorInfo.

Note: OS/400 and z/OS C/C++ do not support IGUIErrorInfo. It is listed here because versions of Open Class Library on other operating systems do support it.

Error Code

Use these member functions to determine which class library an exception originated from.

errorCodeGroup

```
public:ErrorCodeGroup errorCodeGroup() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error group the exception originated from.

setErrorCodeGroup

```
public:IException& setErrorCodeGroup(ErrorCodeGroup errorGroup)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the ID of the originating class library into the exception object.

baseLibrary

```
public:static ErrorCodeGroup const baseLibrary
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the error group for Open Class Library errors.

CLibrary

```
public:static ErrorCodeGroup const CLibrary
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the error group for the C library errors.

operatingSystem

```
public:static ErrorCodeGroup const operatingSystem
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the error group for operating system errors.

other

```
public:static ErrorCodeGroup const other
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the error group for errors which do not fall in any of the other groups.

presentationSystem

```
public:static ErrorCodeGroup const presentationSystem
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Specifies the error group for presentation system errors.

Error Information

Use these member functions to get or modify the error identifier of the exception object.

errorId

```
public:ioc::unumeric_t errorId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error ID of the exception.

setErrorId

```
public:IException& setErrorId(ioc::unumeric_t errorId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the error ID to the specified value.

errorId

The identifier you want to associate with this error.

Exception Location

Use these member functions to set and access the location information in the exception object.

addLocation

```
public:virtual IException&
  addLocation( const IExceptionLocation& location )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the location information to the exception object. The Open Class Library captures this information when an exception is thrown or rethrown. An array of IExceptionLocation objects is stored in the exception object.

location

An IExceptionLocation object containing the following:

- Function name
- File name
- Line number where the function is called.

locationAtIndex

```
public:const IExceptionLocation*
  locationAtIndex( ioc::unumeric_t locationIndex ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the IExceptionLocation object at the specified index.

locationIndex
If the index is not valid, a 0 pointer is returned.

locationCount

```
public:ioc::unumeric_t locationCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of locations stored in the exception location array.

Exception Logging

Use these member functions to log exception information.

logExceptionData

```
public:virtual IException& logExceptionData()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Logs the exception data stored in the IException object using the function specified by IException::setTraceFunction. If you have not set a tracing function, the exception information is written to standard error output.

logExceptionDataW

```
public:virtual IException& logExceptionDataW()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Logs the exception data stored in the IException object using the function specified by IException::setTraceFunction. If you have not set a tracing function, the exception information is written to standard error output.

setTraceFunction

```
public:static IException::TraceFn*
  setTraceFunction( IException::TraceFn& traceFunction )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Registers an object of IException::TraceFn to be used to log exception data. The ITrace member functions and macros write the trace messages. IException::logExceptionData calls IException::TraceFn::write during exception processing to write the data. If you do not register an object, data is written to standard error output.

traceFunction
Your own trace function implementation.

IException::logExceptionDataW (the Unicode version of logExceptionData) calls IException::TraceFn::writeW (the Unicode version of TraceFn::write) which by default converts any Unicode string to char* string before calling TraceFn::write. This ensures that existing TraceFn object still works with Unicode exception data. To change this behavior, you can override the IException::TraceFn::writeW method.

Exception Severity

Use these member functions to set and determine the severity of the error condition.

isRecoverable

```
public:virtual int isRecoverable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the thrower (that is, whatever creates the exception) determines the exception is recoverable, 1 is returned. If the thrower determines it is unrecoverable, 0 is returned.

setSeverity

```
public:IException& setSeverity(Severity severity)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the severity of the exception.

severity

Use the enumeration Severity to specify the severity of the exception.

Exception Text

Use these member functions to set, modify, and retrieve the exception text in the object.

appendText

```
public:IException& appendText(const char* errorText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Appends the specified text to the text string on the top of the exception text stack.

errorText

The text you want to append.

appendTextW

```
public:IException& appendTextW(const ioc::unichar_t* errorText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Appends the specified 16-bit Unicode string to the text string on the top of the exception text stack.

errorText

The text you want to append.

setText

```
public:IException& setText(const char* errorText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the specified text to the top of the exception text stack.

errorText

The error text you want to add.

setTextW

```
public:IException& setTextW(const ioc::unichar_t* errorText)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the specified Unicode string to the top of the exception text stack.

errorText
The error text you want to add.

text

```
public:const char* text(ioc::unumeric_t indexFromTop = 0) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a constant char* pointing to an exception text string from the exception text stack.

indexFromTop
The default index is 0, which is the top of the stack. If you specify an index which is not valid, a 0 pointer is returned.

textCount

```
public:ioc::unumeric_t textCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of text strings in the exception text stack.

textW

```
public:const ioc::unichar_t*
  textW( ioc::unumeric_t indexFromTop = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of text(). Returns a constant char* pointing to an exception text string from the exception text stack.

indexFromTop
The default index is 0, which is the top of the stack. If you specify an index which is not valid, a 0 pointer is returned.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

These member functions support the throwing of exceptions.

assertParameter

```
public:static void
  assertParameter( const char* exceptionText,
                   const IExceptionLocation& location )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The IASSERT macro uses this function to do the following:

1. Create an IAssertionFailure exception
2. Add the location information to it
3. Log the exception data
4. Throw the exception

exceptionText

The text describing the exception.

location

An IExceptionLocation object containing the following:

- Function name
- File name
- Line number where the function is called

assertParameterW

```
public:static void
  assertParameterW( const ioc::unichar_t* exceptionText,
                    const IExceptionLocation& location )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of assertParameter(). The IASSERTW macro uses this function to do the following:

1. Create an IAssertionFailure exception
2. Add the location information to it
3. Log the exception data
4. Throw the exception

exceptionText

The text describing the exception.

location

An IExceptionLocation object containing the following:

- Function name
- File name
- Line number where the function is called

IException - Enumerations

Severity

Use these enumerators to specify the severity of the exception:

unrecoverable

Classifies the exception as unrecoverable.

recoverable

Classifies the exception as recoverable.

IException - Type Definitions

ErrorCodeGroup

```
typedef const char * ErrorCodeGroup
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This identifies the source of the exception's error code.

IException - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IException::TraceFn

Objects of the class IException and its derived classes use IException::TraceFn to log exception object data.

A default TraceFn derived object is registered by the Collection Class Library. If the User Interface Library is used, it registers a TraceFn derived object, which overrides the write function. It uses ITrace to write out the buffers of data, so the buffers will be written to wherever the ICLUI TRACETO environment variable directs the output from ITrace.

If you want to provide your own tracing function, derive your own class from IException::TraceFn and register it with IException using IException::setTraceFunction. You can completely take over exception logging by overriding the logData function. You are passed the IException object so you can completely customize the logging of exception data. If you only want to change how the buffers of exception data are logged, you should override the write function.

The exceptionLogged function is provided so that you can determine when the last buffer of exception data has been passed to the write function by the default logData function. This allows you to gather all of the exception data by only overriding the write and exceptionLogged functions for situations where you must write out all of the exception data with one call.

Class header file: iexcbase.hpp

IException::TraceFn - Hierarchy List

IException::TraceFn

IException::TraceFn - Member Functions and Data by Group

Constructors & Destructor

The only way to create objects of this class is from a derived class. Therefore, all constructors for this class are protected.

Derived classes use these members to create objects of this class.

TraceFn

```
protected:TraceFn()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This default constructor can be used by derived classes to create objects of this class.

Tracing

The IException's logExceptionData member uses these members to log instance data of exception objects.

logData

```
public:virtual void logData(IException& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Logs error information contained in an IException object.

logDataW

```
public:virtual void logDataW(IException& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Logs error information (in Unicode format) contained in an IException object.

write

```
public:virtual void write(const char* buffer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes a buffer of exception data.

writeW

```
public:virtual void writeW(const ioc::unichar_t* buffer)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of write(). Writes a buffer of exception data.

exceptionLogged

```
protected:virtual void exceptionLogged()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is called by the default logData function after the last buffer of exception data has been passed to the write function.

IException::TraceFn - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IExceptionLocation

The IExceptionLocation class saves the location information when an exception is thrown or rethrown. None of the functions in this class throws exceptions because an exception probably has been thrown already or is about to be thrown.

Typically, either the ITHROW or IRETHROW macro creates an IExceptionLocation object when an exception is to be thrown or rethrown, respectively. However, you can create your own IExceptionLocation object by constructing it yourself or by using the IEXCEPTION_LOCATION macro.

Class header file: iexcbase.hpp

IExceptionLocation - Hierarchy List

IExceptionLocation

IExceptionLocation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct objects of this class.

~IExceptionLocation

```
public:~IExceptionLocation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IExceptionLocation

You can create objects of this class by doing the following:

- Using the constructors.
- Using the macro IEXCEPTION_LOCATION. This macro captures the current location information using constants provided by the compiler for all of the parameters. Default values are provided for all the parameters to support environments in which all constants or alternative means for getting location information are not provided.

Overload 1

```
public:IExceptionLocation(const IExceptionLocation& loc)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IExceptionLocation( const ioc::unichar_t* fileName,
                           const ioc::unichar_t* functionName = 0,
                           ioc::unumeric_t lineNumber = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IExceptionLocation( const char* fileName,
                           const char* functionName = 0,
                           ioc::unumeric_t lineNumber = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IExceptionLocation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor for IExceptionLocation().

Assignment

Assignment operator for IExceptionLocation.

operator =

```
public:IExceptionLocation&
  operator =( const IExceptionLocation& loc )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attributes

Use these member functions to return the attributes of the exception location object.

fileName

```
public:const char* fileName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the path-qualified source file name where an exception has been thrown or rethrown.

fileNameW

```
public:const ioc::unichar_t* fileNameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the path-qualified source file name (in Unicode format) where an exception has been thrown or rethrown.

functionName

```
public:const char* functionName() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the function that has thrown or rethrown an exception.

functionNameW

```
public:const ioc::unichar_t* functionNameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the function (in Unicode format) that has thrown or rethrown an exception.

lineNumber

```
public:ioc::unumeric_t lineNumber() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the line number of the statement in the source file from which an exception has been thrown or rethrown.

IExceptionLocation - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IInvalidName

IInvalidName provides an exception reporting that an invalid name, such as a file name or a network resource name, was used for an object. You can use the standard error message the class provides when you throw an exception under these circumstances or you can construct an instance of IInvalidName that allows you to specify a custom message.

Class header file: igenexc.hpp

IInvalidName - Hierarchy List

IException

IInvalidRequest

IInvalidName

IInvalidName - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IInvalidName object from another.

~IInvalidName

```
public:virtual ~IInvalidName()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IInvalidName

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IInvalidName(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an invalid name exception that provides a custom message giving specific details of the error condition.

text The error message to use for this exception. If you do not provide an error message, a default message is used.

Overload 2

```
public:IInvalidName(const IInvalidName&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IInvalidName - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IInvalidRequest			
Definition	**Page Number**	**Definition**	**Page Number**
~IInvalidRequest	781	IInvalidRequest	781
nameW	782		

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IInvalidParameter

The IInvalidParameter class represents an exception. When a member function detects an invalid input parameter, the member function creates and throws an object of the IInvalidParameter class. This exception is identical to the exception IAssertionFailure, with one difference: IInvalidParameter is thrown whether or not you define IC_DEVELOP (when the application is built).

Class header file: iexcbase.hpp

IInvalidParameter - Hierarchy List

IException

IInvalidParameter

IEntityTypeMismatch

IConditionInvalid

IEntityInvalid

IInvalidParameter - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IInvalidParameter object from another.

~IInvalidParameter

```
public:virtual ~IInvalidParameter()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IInvalidParameter

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IInvalidParameter( const char* errorText,
                  ioc::unumeric_t errorId = 0,
                  Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IInvalidParameter( const ioc::unichar_t* errorText,
                  ioc::unumeric_t errorId = 0,
                  Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IInvalidParameter(const IInvalidParameter& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IInvalidParameter - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IInvalidRequest

The IInvalidRequest class represents an exception. Whenever an object cannot satisfy a request, the member function creates and throws an object of the IInvalidRequest class. An example of such a request occurs if you try to paste text from the system clipboard, but the clipboard has no data.

Class header file: iexcbase.hpp

IInvalidRequest - Hierarchy List

IException

IInvalidRequest

IInvalidName

IEndOfStream

IInvalidRequest - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one InvalidRequest object from another.

~IInvalidRequest

```
public:virtual ~IInvalidRequest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IInvalidRequest

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IInvalidRequest( const char* errorText,
                        ioc::unumeric_t errorId = 0,
                        Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IInvalidRequest( const ioc::unichar_t* errorText,
                        ioc::unumeric_t errorId = 0,
                        Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId

The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IInvalidRequest(const IInvalidRequest& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string.. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IInvalidRequest - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768

IException			
Definition	**Page Number**	**Definition**	**Page Number**
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IMessageText

The IMessageText class loads message text from a message file. When the Open Class Library detects an error condition and prepares to throw an exception, the library creates an object of this class if it is using customized message text. You can use the message text provided by this class to construct an object of a class derived from IException.

To override the name of the default message file, you can either:

1. use the setMessageFile method (preferred); OR
2. set the environment variable ICLUI_MSGFILE to the name of the preferred message file.

Changes from IOC V4 to V5: The ″const char *″ operator is now obsoleted. Users are encouraged to use specific methods: ″text″ or ″textW″ to retrieve the message text.

AIX Considerations

The IMessageText object searches for the message file using the NLSPATH environment setting.

OS/400 Considerations

On OS/400 there are both message files and message catalogs. The IMessageText class looks for the specified messageId in a message file on the library list. If the message file or messageId can not be found an attempt is made to find a message catalog with the specified name messageFileName. The message catalog is found by checking the current working directory.

OS/2 Considerations

The IMessageText object searches for the message file as follows:

- The system root directory

- The current working directory
- The DPATH environment setting
- The APPEND environment setting

Typically, message files have the extension .MSG.

Windows Considerations

The IMessageText object uses Win32 API LoadLibrary to load the message resources. LoadLibrary searches resources in the following sequence:

- The directory from which the application loaded.
- The current directory.
- The Windows system directory.
- The Windows directory.
- The PATH environment setting

z/OS Considerations

On z/OS, IMessageText class initially looks for the specified messageId in a Message Table Module residing in the native data set. The message file name passed must be a Language Environment Message Table Module name of the form IxxxMSGT, where xxx is a three-letter facility ID (for example, CLE for class libraries). The Language Environment Message Table Module must be accessible at run time, in the LPA, TSOLIB, and so on. The messageId argument maps to the Language Environment `msgno` tag in the message source file. IMessageText does not use the INCLUDE file generated by CEEBLDTX.

If the message file or messageId cannot be found, an attempt is made to find a message file in the UNIX Systems Services path, given that the z/OS UNIX kernel is available and active. If the message file name passed only contains the file name without the full path, the file name is searched for in the path specified in the environment variable NLSPATH with name substituted for %N. If NLSPATH does not exist in the environment, or if a message catalog cannot be found in any of the components specified by NLSPATH, then the default path of ″/usr/lib/nls/msg/%L/%N″ is used. The ″%L″ component of this default path is replaced by the setting of LC_MESSAGES if the value of oflag is NL_CAT_LOCALE, or the LANG environment variable if the value of oflag is 0.

Class header file: imsgtext.hpp

IMessageText - Hierarchy List

IMessageText

IMessageText - Member Functions and Data by Group

Constructors & Destructor

Use these member function to construct, copy, and destruct objects of this class.

~IMessageText

```
public:~IMessageText()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IMessageText

Constructs an IMessageText object by retrieving a message from a file and optionally inserting additional text strings into the retrieved message.

You can have the object insert the text strings through substitution symbols within the message. For example:

```
The application cannot find the file, %1, at the specified path, %2.
```

You must use the numbers in sequence. For example, you cannot use %1, %2 and %5 in a message, skipping %3 and %4. Instead, you must use %1, %2 and %3. You must specify the substitution symbols sequentially, and the numbers of the text insertion parameters must match their respective substitution symbol.

Overload 1

```
public:IMessageText(ioc::unumeric_t messageId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

messageId
The message ID.

Overload 2

```
public:IMessageText( ioc::unumeric_t messageId,
              const ioc::unichar_t* messageFileName,
              const ioc::unichar_t* textInsert1 = 0,
              const ioc::unichar_t* textInsert2 = 0,
              const ioc::unichar_t* textInsert3 = 0,
              const ioc::unichar_t* textInsert4 = 0,
              const ioc::unichar_t* textInsert5 = 0,
              const ioc::unichar_t* textInsert6 = 0,
              const ioc::unichar_t* textInsert7 = 0,
              const ioc::unichar_t* textInsert8 = 0,
              const ioc::unichar_t* textInsert9 = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

messageId
The message ID.

messageFileName
The name of the message file to retrieve the message from. The message file name must include the file extension. If you specify o, the message text is in a message segment bound to the application and the IMessageText object loads the message from the application. Otherwise, the library searches for the message text in the specified message file.

Note: If the message text cannot be retrieved from the message file, the following default text is used: ″Unable to load text from message file.″

textInsert1 through textInsert9
The substitution text to be inserted in place of the substitution symbol in the message.

Overload 3

```
public:IMessageText( ioc::unumeric_t messageId,
              const char* messageFileName,
              const char* textInsert1 = 0,
              const char* textInsert2 = 0,
              const char* textInsert3 = 0,
              const char* textInsert4 = 0,
```

```
                    const char* textInsert5 = 0,
                    const char* textInsert6 = 0,
                    const char* textInsert7 = 0,
                    const char* textInsert8 = 0,
                    const char* textInsert9 = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

messageId
The message ID.

messageFileName
The name of the message file to retrieve the message from. The message file name must include the file extension. If you specify o, the message text is in a message segment bound to the application and the IMessageText object loads the message from the application. Otherwise, the library searches for the message text in the specified message file.

Note: If the message text cannot be retrieved from the message file, the following default text is used: ″Unable to load text from message file.″

textInsert1 through textInsert9
The substitution text to be inserted in place of the substitution symbol in the message.

Overload 4

```
public:IMessageText(const IMessageText& text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

You can construct objects of this class using the copy constructor.

text The error message text.

Assignment

Use this assignment operator to assign one IMessageText object to another.

operator =

```
public:IMessageText& operator =(const IMessageText& text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns another message object to this object.

Message File Support

Use these member functions to get or set the message file name.

messageFile

```
public:static const char* messageFile()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the message file name used to extract the message text. You can change the message file name using the setMessageFile() method. The IMessageText(unsigned long messageId) constructor also uses this method to retrieve the message file name.

messageFileW

```
public:static const ioc::unichar_t* messageFileW()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the message file name in Unicode format.

setMessageFile

```
public:static void setMessageFile(const char* msgFileName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets a new message file from which the message text will be extracted. Since only one message file can be active at a time, the Open Class Library message text will also be loaded from the new message file.

setMessageFileW

```
public:static void
  setMessageFileW( const ioc::unichar_t* msgFileName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Set the message file name with a Unicode string.

Text Operations

Use these member functions to obtain the text from the object and to set the default text for the object.

setDefaultText

```
public:IMessageText& setDefaultText(const char* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the default message text to the specified text string. The text is set only if the constructor cannot load the text for the specified message ID.

Note: The default text is: ″Unable to load text from message file.″

text The new default text string.

setDefaultTextW

```
public:IMessageText& setDefaultTextW(const ioc::unichar_t* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the default text string with a Unicode string.

successful

```
public:bool successful() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Indicates whether the message extraction was successful.

text

```
public:const char* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the message text.

textW

```
public:const ioc::unichar_t* textW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the message text in Unicode format.

IMessageText - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMustBeEmpty

The IMustBeEmpty class provides a message reporting that the function could not perform an operation on a container because the container is not empty. When your code encounters circumstances that cause it to throw this exception, you can use an instance of this class to provide the standard error message or you can construct an object of this class that contains a custom message.

Class header file: igenexc.hpp

IMustBeEmpty - Hierarchy List

IException

IAccessError

IMustBeEmpty

IMustBeEmpty - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IMustBeEmpty object from another.

~IMustBeEmpty

```
public:virtual ~IMustBeEmpty()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IMustBeEmpty

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IMustBeEmpty(const IMustBeEmpty&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IMustBeEmpty(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an exception that allows you to specify a customized error message. If you provide one, your error message is used instead of the default one.

text The error message to use for this exception. If no error message is provided, a default message will be used instead.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IMustBeEmpty - Inherited Member Functions and Data

Inherited Public Functions

IAccessError			
Definition	**Page Number**	**Definition**	**Page Number**
~IAccessError	731	IAccessError	731
nameW	732		

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IObjectNotFound

The IObjectNotFound class provides a message reporting that an operation failed because it was unable to locate the requested object. You can use an instance of this class to signal that your code encounted circumstances causing it to throw the exception. If you want to provide more specific information, IObjectNotFound allows you to construct an error message and use it instead of the default one.

Class header file: igenexc.hpp

IObjectNotFound - Hierarchy List

IException

IAccessError

IObjectNotFound

IObjectNotFound - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IObjectNotFound object from another.

~IObjectNotFound

```
public:virtual ~IObjectNotFound()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IObjectNotFound

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IObjectNotFound(const IObjectNotFound&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Overload 2

```
public:IObjectNotFound(const char* text = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an exception that allows you to specify a customized error message. If you provide one, your error message is used instead of the default one.

text The customized error message to use for this exception. If you do not provide an error message, a default one is used.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IObjectNotFound - Inherited Member Functions and Data

Inherited Public Functions

IAccessError			
Definition	**Page Number**	**Definition**	**Page Number**
~IAccessError	731	IAccessError	731
nameW	732		

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769

IException			
Definition	**Page Number**	**Definition**	**Page Number**
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IOutOfMemory

The IOutOfMemory class represents an exception. The Open Class Library installs its own new_handler() and throws an IOutOfMemory exception instead of an std::bad_alloc exception when the application runs out of memory. This is a deviation from the ANSI standard.

Class header file: iexcbase.hpp

IOutOfMemory - Hierarchy List

IException

IResourceExhausted

IOutOfMemory

IOutOfMemory - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IOutOfMemory object from another.

~IOutOfMemory

```
public:virtual ~IOutOfMemory()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IOutOfMemory

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IOutOfMemory( const ioc::unichar_t* errorText,
              ioc::unumeric_t errorId = 0,
              Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IOutOfMemory( const char* errorText,
              ioc::unumeric_t errorId = 0,
              Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IOutOfMemory(const IOutOfMemory& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IOutOfMemory - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IResourceExhausted			
Definition	**Page Number**	**Definition**	**Page Number**
~IResourceExhausted	798	IResourceExhausted	798

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IOutOfSystemResource

The IOutOfSystemResource class represents an exception. When a member function makes an operating system resource request that the system cannot satisfy, the member function creates and throws an object of the IOutOfSystemResource class.

Class header file: iexcbase.hpp

IOutOfSystemResource - Hierarchy List

IException

IResourceExhausted

IOutOfSystemResource

IOutOfSystemResource - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IOutOfSystemResource object from another.

~IOutOfSystemResource

```
public:virtual ~IOutOfSystemResource()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IOutOfSystemResource

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IOutOfSystemResource( const char* errorText,
                             ioc::unumeric_t errorId = 0,
                             Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText

The text describing this particular error.

errorId

The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IOutOfSystemResource( const ioc::unichar_t* errorText,
                     ioc::unumeric_t errorId = 0,
                     Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText

The text describing this particular error.

errorId

The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IOutOfSystemResource(const IOutOfSystemResource& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IOutOfSystemResource - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IResourceExhausted			
Definition	**Page Number**	**Definition**	**Page Number**
~IResourceExhausted	798	IResourceExhausted	798

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IResourceExhausted

The IResourceExhausted class represents an exception. When a member function makes a resource request of the operating system or the presentation system that it cannot satisfy, the member function creates and throws an object of the IResourceExhausted class or one of its derived classes. IResourceExhausted is the generic out-of-resource class. Member functions use IResourceExhausted whenever its derived classes, which are for specific out-of-resource cases, are not applicable.

The derived classes for IResourceExhausted are the following:

IOutOfMemory

IOutOfSystemResource

IOutOfWindowResource

Class header file: iexcbase.hpp

IResourceExhausted - Hierarchy List

IException

IResourceExhausted

IOutOfSystemResource

IOutOfMemory

IOutOfWindowResource

IResourceExhausted - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IResourceExhausted object from another.

~IResourceExhausted

```
public:virtual ~IResourceExhausted()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IResourceExhausted

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IResourceExhausted( const char* errorText,
                           ioc::unumeric_t errorId = 0,
                           Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IResourceExhausted( const ioc::unichar_t* errorText,
                    ioc::unumeric_t errorId = 0,
                    Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IResourceExhausted(const IResourceExhausted& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

Use these member functions to determine the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IResourceExhausted - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771

IException			
Definition	**Page Number**	**Definition**	**Page Number**
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IStartStopTimingTest

IStartStopTimingTest is an abstract framework class for timing measurements too complicated to do with ITimingTest. It is a subclass of ITimingTest with the following:

1. Lower precision
2. The ability to start and stop the timer.

It is used when the timer needs to be started and stopped multiple times. Because this results in a loss of precision, use ITimingTest whenever possible.

To use, subclass and override startStopTimingTest much as you would override ITimingTest::timingTest.

Class header file: issttest.hpp

IStartStopTimingTest - Hierarchy List

ITest

ITimingTest

IStartStopTimingTest

IStartStopTimingTest - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~IStartStopTimingTest

```
public:virtual ~IStartStopTimingTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs objects of this class.

IStartStopTimingTest

Overload 1

```
public:IStartStopTimingTest(const IStartStopTimingTest& other)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor that accepts the IStartStopTimingTest to be copied.

Overload 2

```
protected:IStartStopTimingTest( ioc::unumeric_t timingCount,
                      ioc::unumeric_t sampleCount )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor that explicitly sets timing count and sample count.

timingCount
Explicit timing count.

sampleCount
Explicit sample count.

Overload 3

```
protected:IStartStopTimingTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor.

Assignment Operators

Standard assignment operators.

operator =

```
public:IStartStopTimingTest&
  operator =( const IStartStopTimingTest& other )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator which accepts the IStartStopTimingTest object used in the assignment.

Specialization Methods

Advanced specialized subclass methods to be used only by those users with a good understanding of the Timing Test classes.

doUntimedTest

```
protected:virtual void doUntimedTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Runs the startStopTimingTest without performing the timing. For example, execute only timingSetup, startStopTimingTest, and timingCleanup. Useful to debug the startStopTimingTest without concern for the timing aspects.

oneEmptySample

```
protected:virtual double oneEmptySample()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Runs an empty sample. Useful to determine the amount of time necessary for the overhead of doing the timing.

oneSample

```
protected:virtual double oneSample()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines the time it takes to run one sample.

samplingErrorMicroseconds

```
protected:virtual double samplingErrorMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the sampling error.

For start-stop tests, the sampling error per sample is k alpha, where k is the number of calls to start or stop per call to startStopTimingTest.

Timing Methods

Methods used to start and stop the timing test.

emptyStartStopTimingTest

```
protected:virtual void
  emptyStartStopTimingTest( ITimingTestStopwatch& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This class calls the timer repeatedly to calibrate times and account for the overhead involved in starting or stopping the timer. If your startStopTimingTest method starts and stops the timer more than once, override emptyStartStopTimingTest. In your version, start and stop the timer as many times as you do in startStopTimingTest. The class will use this method to calibrate times and to account for the overhead involved in starting or stopping the timer.

The ITimingTestStopwatch is the stopwatch object which you should start and stop. In your overridden version, start and stop this object as many times as you do in startStopTimingTest. For example: Start, stop, start, stop. IStartStopTimingTest uses this function to calibrate times and account for the overhead involved in starting or stopping the timer.

startStopTimingTest

```
protected:virtual void startStopTimingTest(ITimingTestStopwatch&) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method is analogous to ITimingTest::timingTest. Subclasses override this method to do the following:

- Do necessary setup
- Start the timer (which is passed into the method)
- Do the timed operation
- Stop the timer
- Do necessary cleanup

For example:

```
{
  // Setup the test
  timer.start();
  // Do the operation to be timed
  // ...
  // Stop the timer
  timer.stop();
  // Cleanup
};
```

If necessary, the timer can be started and stopped more than once, but this should be avoided if at all possible.

The startStopTimingTest method should always assume the timer is stopped on entry into the method, and should always stop the timer before exiting the method.

IStartStopTimingTest - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInputs	811
exceptionStage	812	lookupInfo	816
operator =	812	outputTextStream	816
run	812	runCount	817
setInputs	811	success	813

ITimingTest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITimingTest	835	automaticSampleTolerance	836
automaticSampling	836	calibrate	838
calibratedMeanMicroseconds	838	calibratedMedianMicroseconds	838
clockNowOverhead	838	continualCalibration	838
copyInfo	838	copySamples	838
correlationMicroseconds	839	correlationTooLarge	839
errorMicroseconds	839	errorTooLarge	839
ITimingTest	835	maximumAllowedMedianMicroseconds	839
maximumSampleWindows	839	minimumSampleCorrelation	836
operator =	836	relativeErrorMicroseconds	839

ITimingTest			
Definition	**Page Number**	**Definition**	**Page Number**
sampleCount	839	samplesDiscarded	839
samplesTaken	840	setAutomaticSampleTolerance	837
setAutomaticSampling	837	setContinualCalibration	840
setMaximumAllowedMedianMicroseconds	840	setMaximumSampleWindows	837
setMinimumSampleCorrelation	837	setSampleCount	840
setTimingCount	840	setTimingOn	841
timingCount	841	timingOn	841

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

ITimingTest			
Definition	**Page Number**	**Definition**	**Page Number**
kDefaultAutomaticSamplingTolerance	837	kDefaultMinimumCorrelation	837

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	isReset	817
isSuccessUndecided	818	ITest	812
print	814	reset	814
setStickySuccess	818	setSuccess	818
test	815		

ITimingTest			
Definition	**Page Number**	**Definition**	**Page Number**
cleanup	841	ITimingTest	835
print	841	setup	841
test	841	timingCleanup	842
timingSetup	842	timingTest	842

Inherited Protected Data

None

IStreamTestOf

Tests the streaming of the target object using operator<<= and operator>>= .

IStreamTestOf is a subclass of ITest which tests the streaming of the target. Clients should create a subclass of IStreamTestOf or use streamTestMacro (see below).

This test is a good way to leverage testing effort since it allows one to test classes if those classes are expected to adhere to some protocol. This protocol test is implemented using a template class. Target classes MUST have operator= defined to be tested by IStreamTestOf. One macro and one global function are provided for easy use of this class.

The macro streamTestMacro does everything you need in one line. It writes the entire header and the entire implementation. Just supply the name of the test class, target class and the parameter list for the constructor that creates the target. The parameter list may be left blank to indicate that the empty constructor is to be used to create the target.

For example, ISample is the target class that we want to test streaming. ISample has operator<<=, operator>>=, and operator== defined.

```
streamTestMacro(ISampleTest,ISample,(1));
```

The following is an example of a class which creates the target object using the empty constructor:

```
streamTestMacro(ISampleTest,ISample,);
```

Now you can run ISampleTest with runTestImplementationMacro:

```
runTestImplementationMacro(ISampleTest);
```

Or you can use this global function which creates and returns an instance of IStreamTestOf, as follows:

```
createStreamTest(new ISample(2));
```

This can be used with the adoptTest method in the ITestCollection class.

Class header file: istrmtst.hpp

IStreamTestOf - Hierarchy List

ITest

IStreamTestOf

IStreamTestOf - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~IStreamTestOf

```
public:virtual ~IStreamTestOf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs objects of this class.

IStreamTestOf

Constructs objects of this class.

Overload 1

```
public:IStreamTestOf(const IStreamTestOf < AType >&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Overload 2

```
public:IStreamTestOf(AType* target)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor which specifies the target object to perform tests on.

Assignment Operators

Use this operator to replace the existing object with the assigned, given one.

operator =

```
public:IStreamTestOf < AType >&
  operator =( const IStreamTestOf < AType >& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator.

Meta Information

Class member functions used to provide information about the tests.

copyInfo

```
public:virtual void copyInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Within copyInfo, ISteamTestOf calls the inherited implementation. For example, IDerived::copyInfo should call IBase::copyInfo. Then copyInfo should add key/value pairs to infoMap by calling the addInfo() method.

Output Methods

Class members used to display information to the user.

targetClassNameAsString

```
protected:virtual const char* targetClassNameAsString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the target class. Be default, it returns null. You can override this method if the target class name is to be reported in the test output.

Test Methods

Methods used to run the streaming test.

test

```
protected:virtual void test()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams the target object out to and in from the memory stream and compares the streamed-in object with the original.

The test method also verifies that the stream position is consistent after stream-out and stream-in.

Note: Do not override this method. The best way to use the IStreamTestOf framework is to use the protocol StreamTestMacro.

ITest overrides; these methods are called by the framework ITest::run().

IStreamTestOf - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInputs	811
exceptionStage	812	lookupInfo	816
operator =	812	outputTextStream	816
run	812	runCount	817
setInputs	811	success	813

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	cleanup	814
isReset	817	isSuccessUndecided	818
ITest	812	print	814
reset	814	setStickySuccess	818
setSuccess	818	setup	814
test	815		

Inherited Protected Data

None

ISystemErrorInfo

The ISystemErrorInfo class represents error information that you can include in an exception object. When an operating system call results in an error condition, objects of the ISystemErrorInfo class are created. You can use the error text to construct a derived class object of IException

The Open Class Library provides the ITHROWSYSTEMERROR macro for throwing exceptions constructed with the following ISystemErrorInfo information:

- The error ID returned from the system function
- The name of the system function that returned an error code
- One of the values of the enumeration IBaseErrorInfo::ExceptionType, which specifies the type of exception this macro creates
- One of the values of the enumeration IException::Severity, which specifies the severity of the exception

This macro generates code that calls throwSystemError, which does the following:

1. Creates an ISystemErrorInfo object
2. Uses the object to create an IException object
3. Adds the operatingSystem error group to the object
4. Adds location information
5. Logs the exception data
6. Throws the exception

If Unicode error info is required, use the ITHROWSYSTEMERRORW macro. The macro generates code that calls throwSystemErrorW, which logs the exception data in Unicode character string.

AIX Considerations

You can create objects of this class on AIX, but the objects contain no useful information and only have the default message: ″System exception condition detected.″

OS/400 Considerations

OS/400 C/C++ does not support the ISystemErrorInfo class. It is described here because versions of the Open Class Library on other operating systems do support it.

OS/2 Considerations

ISystemErrorInfo is used to obtain the error information for OS/2 system calls.

z/OS Considerations

You can create objects of this class on z/OS C/C++, but the objects contain no useful information and only have the default message: ″System exception condition detected.″

Class header file: iexcbase.hpp

ISystemErrorInfo - Hierarchy List

IBaseErrorInfo

ISystemErrorInfo

ISystemErrorInfo - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~ISystemErrorInfo

```
public:virtual ~ISystemErrorInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ISystemErrorInfo

Constructs an ISystemErrorInfo object.

Overload 1

```
public:ISystemErrorInfo( ioc::unumeric_t systemErrorId,
                         const char* systemFunctionName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ISystemErrorInfo( ioc::unumeric_t systemErrorId,
                         const ioc::unichar_t* systemFunctionName )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ISystemErrorInfo(ioc::unumeric_t systemErrorId)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Error Information

Use these member functions to return error information provided by objects of this class.

errorId

```
public:virtual ioc::unumeric_t errorId() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error ID.

isAvailable

```
public:virtual bool isAvailable() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the error information is available, true is returned.

text

```
public:virtual const char* text() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the error text.

textW

```
public:virtual const ioc::unichar_t* textW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return the error text in Unicode format.

Throw Support

Use these member functions to support the throwing of exceptions using information from an ISystemErrorInfo object. The throwSystemError function is used by the ITHROWSYSTEMERROR macro.

throwSystemError

```
public:static void
  throwSystemError( ioc::unumeric_t systemErrorId,
                    const char* functionName,
                    const IExceptionLocation& location,
                    IBaseErrorInfo::ExceptionType name = accessError,
                    IException::Severity severity = IException::recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Used by the ITHROWSYSTEMERROR macro, this function creates an ISystemErrorInfo object and uses the text from it to do the following:

1. Create an exception object
2. Add the location information to it
3. Log the exception data
4. Throw the exception

systemErrorId
The error ID from the system.

functionName
The name of the function where the exception occurred.

location
An IExceptionLocation object containing the following:
- Function name
- File name
- Line number where the function is called

name Use the enumeration IBaseErrorInfo::ExceptionType to specify the type of the exception. The default is accessError.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

throwSystemErrorW

```
public:static void
  throwSystemErrorW( ioc::unumeric_t systemErrorId,
                     const ioc::unichar_t* functionName,
                     const IExceptionLocation& location,
                     IBaseErrorInfo::ExceptionType name = accessError,
                     IException::Severity severity = IException::recoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Unicode version of throwSystemError(). Used by the ITHROWSYSTEMERRORW macro, this function creates an ISystemErrorInfo object and uses the text from it to do the following:

1. Create an exception object
2. Add the location information to it
3. Log the exception data
4. Throw the exception

systemErrorId
The error ID from the system.

functionName
The name of the function where the exception occurred.

location
An IExceptionLocation object containing the following:
- Function name
- File name
- Line number where the function is called

name Use the enumeration IBaseErrorInfo::ExceptionType to specify the type of the exception. The default is accessError.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is recoverable.

ISystemErrorInfo - Inherited Member Functions and Data

Inherited Public Functions

IBaseErrorInfo			
Definition	**Page Number**	**Definition**	**Page Number**
~IBaseErrorInfo	746	errorId	747
IBaseErrorInfo	746	isAvailable	747
text	747	throwError	747
throwErrorW	748		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITest

General purpose abstract class for writing Test Framework tests.

Use ITest to derive subclasses for testing features of your test targets. ITest provides an output stream in which you can record any test output. ITest provides input text arguments that you can use to parameterize the kind of test to be performed. Meta-information on the test may also be defined and obtained using the copyInfo and lookupInfo methods.

Class header file: itest.hpp

ITest - Hierarchy List

ITest
- ITestMultiplexer
- ITestCollection
- ITimingTest
- IComparisonTestOf

ITest - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITest

```
public:virtual ~ITest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITest destructor. It does not delete the target. Subclasses must handle the target themselves.

ITest

Constructors.

Overload 1

```
protected:ITest(const ITest& test)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor which creates an ITest object from an existing one.

Overload 2

```
protected:ITest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor to create an ITest object.

Assignment Operators

Use this operator to assign an ITest class to another.

operator =

```
public:ITest& operator =(const ITest& test)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator for ITest.

Exceptions

Class exception member functions used to determine the status of a test run.

exceptionStage

```
public:virtual EStage exceptionStage() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A public exception method to determine when an exception occurred in a run.

Execution

Class member functions used to execute the test.

run

```
public:virtual void run()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method is the public means of telling the test to execute itself. You can call run to execute a test instance.

Derived classes should not override this function. The run function calls following the framework methods:

- reset (if run has been called before)
- setup
- test (to actually perform the test)
- cleanup

Each time run is called, it increments the value returned by runCount.

Execution Results

Class member functions used to determine the status of the test.

success

```
public:bool success() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method returns the success state of the last run. You may call success after a test has been run to see if the test succeeded. This value is not meaningful if the data member isReset is true.

Input Configuration

Manipulate the test's input arguments.

copyInputs

```
public:void copyInputs(int& argc, char * *& argv) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Create a copy of the current input arguments.

Copies parameters that take UNIX-style parameters to argv.
argc UNIX-style argc parameter
argv UNIX-style argument vector

Derived classes should not override this function.

You can call copyInputs after running a test to correlate inputs with the test results. The caller owns the string array and should delete the storage.

setInputs

```
public:void setInputs(const int argc, const char * const* argv)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes current inputs and creates IString objects for each item in argv and adds the text to the inputs.

Sets up new input values. Copies UNIX-style input text to internal storage.
argc UNIX-style argc parameter
argv UNIX-style argument vector

Main Testing Functions

Main member functions used to setup the actual tests.

addInfo

```
protected:virtual void addInfo(const IString&, const IString&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function adds a key/value pair of IString values to the test meta information. These are most often used when overriding the copyInfo member function.

cleanup

```
protected:virtual void cleanup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework method that derived classes override to restore the system to its previous state.

Any actions within setup which need to be undone (for example, storage allocation), should be undone here.

ITest::run always calls cleanup, even if an exception occurs.

print

Allows customization of the way a test prints.

```
protected:virtual void print(ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows customization of the way a test prints by printing to an ITieredTextBuffer object. Derived classes may override this function to customize the way a test prints via operator<< (ITieredTextBuffer).

In the subclass implementation, be sure to call the print method in the immediate base class before doing anything else. Then output your own information with a final carriage return. If a subclass overrides print, this additional information is included in the test result.

Note: The content of the print method is printed out only when you set (through the command line) the echotier level equal to or below ITieredTextBuffer::kDetail.

reset

```
protected:virtual void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the internal state of ITest back to the initial state.

Overriding methods should call the inherited method first. This method is used to perform necessary re-initialization operations for tests that can be run more than once. ITest::run calls this method before running a test a second or later time. ITest::run does not call reset before running a test for the first time. Derived classes that should not be run more than once, can use this method to ensure that multiple calls to run do no harm.

setup

```
protected:virtual void setup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework method that subclasses override to do whatever setup work the test will need.

Subclasses may override setup to perform any appropriate initialization before test is called. For example, a test might instantiate a target object to be tested if there is not one already.

Note: Anything that is done in setup should be undone, if necessary, in cleanup.

test

```
protected:virtual void test() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The framework method that subclasses override to actually perform the test.

The only ITest method that subclasses are required to call inside a test is setSuccess.

Note: Derived classes must override this method, even if they do not override any of the other methods in this class.

Meta-information

Member functions used to setup or retrieve test meta information.

copyInfo

```
public:virtual void copyInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method is used to store meta-information about the client test. Derived classes should override this function.

You can specify various types of meta-information, including the purpose of a test and the name of the class being tested. You can retrieve and display this information by using the -info option when you start RunTest.

When you assign information to a test, associate each test class with key/value pairs in a dictionary. This supports categorization of different kinds of tests and analysis of large numbers of test results.

The text values for the keys can be used with the -info option of RunTest to query a specific key/value pair in the dictionary. The following shows the predefined keys and values:

```
    KEY             KEYWORD(VALUE)        DESCRIPTION
---------------     --------------        -------------------------------------------
kDescriptionKey     Description           Description of this test subclass
kInputSyntaxKey     Input Syntax          Input syntax for text arguments
kTargetClassKey     Target Class          Name of target class
kTestNameKey        Test Name             Name of test
kTestVersionKey     Test Version          Version number of test
kTestTypeKey        Test Type             Type of test (eg. unit, regression, etc.)
kComponentKey       Component             Name of component to which this test applies
```

Inside of copyInfo, derived classes should call the inherited implementation (IDerived::copyInfo should call IBase::copyInfo). Then copyInfo should add key/value pairs to infoMap by calling the addInfo method.

lookupInfo

```
public:virtual bool lookupInfo(const IString& key, IString& info)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Looks for the information corresponding to the given key. If found, it copies the information into the info parameter and returns true. Otherwise, lookupInfo leaves info untouched and returns false. Derived classes should not override this function.

key IString for the key

info IString for returned value

addInfo

```
protected:virtual void addInfo(const IString&, const IString&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member function adds a key/value pair of IString values to the test meta information. These are most often used when overriding the copyInfo member function.

Output

Member functions used by the tests to display output.

outputTextStream

```
public:ITieredTextBuffer& outputTextStream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to a ITieredTextBuffer object. Derived classes may use the returned reference transparently to produce textual output.

print

Allows customization of the way a test prints.

```
protected:virtual void print(ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows customization of the way a test prints by printing to an ITieredTextBuffer object. Derived classes may override this function to customize the way a test prints via operator<< (ITieredTextBuffer).

In the subclass implementation, be sure to call the print method in the immediate base class before doing anything else. Then output your own information with a final carriage return. If a subclass overrides print, this additional information is included in the test result.

Note: The content of the print method is printed out only when you set (through the command line) the echotier level equal to or below ITieredTextBuffer::kDetail.

Public Data Members

Public data members used to store meta-information about this test. See the member functions copyInputs, setInputs, copyInfo, lookupInfo, and addInfo.

kComponentKey

```
public:static const IString & kComponentKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Name of component that this test applies to.

kDescriptionKey

```
public:static const IString & kDescriptionKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Description of this test subclass.

kInputSyntaxKey

```
public:static const IString & kInputSyntaxKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Input syntax for text arguments.

kTargetClassKey

```
public:static const IString & kTargetClassKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Name of the target class.

kTestNameKey

```
public:static const IString & kTestNameKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The name of the test.

kTestTypeKey

```
public:static const IString & kTestTypeKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A type of test. For example, unit or regression test.

kTestVersionKey

```
public:static const IString & kTestVersionKey
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Version number of the test.

Test Control

Member functions used to control the test status keeping.

runCount

```
public:virtual ioc::unumeric_t runCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of times run has been called.

isReset

```
protected:virtual bool isReset() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the test has been reset without running.

isSuccessUndecided

```
protected:virtual bool isSuccessUndecided() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if setSuccess has not been called or if reset has been called, otherwise returns false.

setStickySuccess

```
protected:virtual void setStickySuccess(bool success)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Used to preserve the failure state of a test until the next run of the test.

The difference between setStickySuccess and setSuccess is that setStickySuccess is called to preserve the failure of a test until the next run of the test. Once setStickySuccess is called with false, the success value will remain false from that point on. It can no longer be changed to true even if setSuccess or setStickySuccess is called with true. If a test is run more than once, the "sticky" flag will be reset in ITest:reset so that success can be set to any value in the next run of the test.

Derived classes can call setStickySuccess at anytime, but especially during test to indicate the test's success.

setSuccess

Sets the success state of a test.

```
protected:virtual void setSuccess(bool success)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subclasses call setSuccess at anytime, but especially during test to indicate the test's success. You can call setSuccess more than once. For example, a derived class might call setSuccess(true) in setup, but then call setSuccess(false) in test.

ITest - Enumerations

EStage

Possible exception value settings are:

kNone
Exception did not occur

kRun Exception in run() but not in any of the following:

kReset
exception in reset()

kSetup
Exception in setup()

kTest exception in test()

kCleanup
Exception in cleanup()

ITest - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITestCollection

ITestCollection is a subclass of ITest which tests a sequence of subtests.

This class allows the test authors to group related ITest subclasses together within a single test. The tests in the collection are run sequentially to determine the success or failure of the entire group. The resulting test passes if all of its subtests pass, and fails otherwise. The subtests are always executed in a fixed order (the order in which subtests were added). The subtests can be run in random order if you call the ″randomlyReorder″ method after adding subtests. ITestCollection does not halt on subtest failure. It runs all the subtests by default. To change this default behavior, use the ″setHaltOnFail(true)″ method in your setupSubtests.

Clients should derive a subclass and override setupSubtests method. In setupSubtests, clients should call ″adoptTest″ for each subtest.

Class header file: itestcol.hpp

ITestCollection - Hierarchy List

ITest

ITestCollection

ITestCollection - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITestCollection

```
public:virtual ~ITestCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITestCollection destructor does not delete the subtests of this collection. Subclasses are responsible for doing this.

ITestCollection

Overload 1

```
public:ITestCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Overload 2

```
public:ITestCollection(const ITestCollection&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Assignment Operators

Use this operator to replace the existing object with the assigned, given one.

operator =

```
public:ITestCollection& operator =(const ITestCollection&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator.

Collection Test Setup

Class members used to set up the collection of tests.

adoptTest

```
public:virtual void adoptTest(ITest* subTest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the subTest to this test's collection of subtests.

The caller must not delete the subTest after calling adopt. ITestCollection takes ownership of the subTest and will manage storage.

propagateInputs

```
protected:virtual void propagateInputs(ITest& subtest)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Framework method which controls how input parameters to the test are sent to subtests. The ITestCollection implementation copies all input arguments unchanged to all subtests.

Called by setup.

Derived classes can override this method to alter this behavior. For example, subclasses may wish to parse or process the inputs before passing them to subtests, or generate inputs for the subtests.

subtest
Subtest to propagate inputs to

setupSubtests

```
protected:virtual void setupSubtests()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is called by setup to create the subtests. Derived classes should

override setupSubtests to instantiate all subtests and call adoptTest for each subtest. Do not delete your subtests after calling adoptTest.

Input and Output

Class members used to display information to the user.

print

```
protected:virtual void print(ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Customized operator<<(ITieredTextBuffer) implementation.

First calls ITest::print(out), and then prints the number of subtests and the current setting of haltOnFail.

ITieredTextBuffer
Tiered text buffer to print to.

Meta Information

Class member functions used to provide information about the tests.

copyInfo

```
public:virtual void copyInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Framework method that derived classes can override to provide meta-information about themselves.

Test Control

Class member functions used to control how to run the collection of tests.

haltOnFail

```
public:bool haltOnFail() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current value that indicates whether or not to halt on failure (the current setting of setHaltOnFail).

randomlyReorder

```
public:virtual void
  randomlyReorder( ioc::numeric_t initialSeed = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Randomly rearranges the order in which tests will be executed by the test method.

initialSeed
Value used for random number generation; default of 0.

setHaltOnFail

```
public:void setHaltOnFail(bool)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Method used to stop running the test collection when any subtest fails.

If client calls this function with true, ITestCollection will halt when any of its subtests fails. If the client calls this function with false, ITestCollection will continue to run all tests, even if one or more of the subtests fail. The default behavior is run all tests.

Test Methods

ITest overridden member function used when setting up and running the collection of tests.

reset

```
protected:virtual void reset()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITestCollection overrides the ITest::reset method to reset all of its subtests as well as itself.

Subclasses should not need to override reset.

setup

```
protected:virtual void setup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITestCollection overrides the ITest::setup method.

ITestCollection calls setupSubtests to create all the subtests. setupSubtests is called only once when setup is called for the first time. Next, it calls propagateInputs for all subtests. It then calls ITest::setup.

Subclasses should not need to override setup.

test

```
protected:virtual void test()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITestCollection overrides the ITest::test method to call the run method for each of its subtests. If all subtests pass, then the ITestCollection passes. If any subtest fails, then the ITestCollection fails. If haltOnFail returns true then no further subtests will be run once any subtest fails. Otherwise, all subtests are always run.

Subclasses should not override test.

ITestCollection - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInputs	811
exceptionStage	812	lookupInfo	816
operator =	812	outputTextStream	816
run	812	runCount	817
setInputs	811	success	813

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	cleanup	814
isReset	817	isSuccessUndecided	818
ITest	812	setStickySuccess	818
setSuccess	818	test	815

Inherited Protected Data

None

ITestMultiplexer

ITestMultiplexer is a subclass of ITest which tests a number of decision functions.

ITestMultiplexer supports the model of multiple decision functions applied to a single test target. Its purpose is to allow multiple related tests to be implemented in one class. Tests are implemented as member functions. Clients derive new subclasses of ITestMultiplexer containing member functions called decision functions which return boolean success values.

ITestMultiplexer selects one or more decision functions to execute in order to perform the test. The selection of the decision function can be set either programmatically or by parsing the test input arguments. To subclass ITestMultiplexer, clients write the decision functions and override loadDecisions(..).

These decision functions are methods which are analogous to the ITest::test method. They are different in that they return a boolean result rather than calling ITest::setSuccess. A TRUE return value indicates success. Each decision function has an associated test key. This key is used to select the decision function at run-time.

The key may be changed in a number of ways. There are methods which directly set the key. Alternatively, a client may specify that one or more of the initial input arguments are to be interpreted as a key. These two methods are mutually exclusive; only one or the other is in effect at any time.

One key comes predefined. This is the key "kAllDecisions", which has the effect of running all decision functions and doing a logical AND of the results. If any decision function fails, then the entire test is considered to fail. Subclasses initialize the key-decision table dynamically. The method loadDecision is a framework method which subclasses MUST override to add each key-decision pair to a dictionary.

Class header file: itestmul.hpp

ITestMultiplexer - Hierarchy List

ITest

ITestMultiplexer

ITestMultiplexer - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITestMultiplexer

```
public:virtual ~ITestMultiplexer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor.

ITestMultiplexer

Overload 1

```
protected:ITestMultiplexer( const IString& theKey,
                  unsigned int howManyInputsAsKeys = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor taking the key and the number of inputs as keys.

theKey
The decision key used for the current key.

howManyInputsAsKeys
The number of inputs that are interpreted as keys.

Overload 2

```
protected:ITestMultiplexer(const ITestMultiplexer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Overload 3

```
protected:ITestMultiplexer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Assignment Operators

Use this operator to replace the existing object with the assigned, given one.

operator =

```
public:ITestMultiplexer& operator =(const ITestMultiplexer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator. The parameter copyIn specifies the ITestMultiplexer object used to do the assignment.

Client Action Functions

Class member functions used to control the decision test functions to the test suite.

addDecision

```
protected:virtual void addDecision(const IString&, ITestDecisionFn)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the pair aKey-aFunc to the dictionary.

Subclasses should call addDecision in their loadDecisions method to register each decision function which they have defined or overridden.

Subclasses should register overridden decision functions with the same key that the base class uses.

key The key to add to the dictionary.

ITestDecisionFn fun
The decision function to add.

andAllDecisions

```
protected:virtual bool andAllDecisions()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called by ITest if the key kAllDecisions is in effect.

Calls all decision functions (including those in base classes which have not been overridden), performs logical AND of their return values, and returns the result. You should not call this method directly. Override this function if you need to change its behavior.

handleDecision

```
protected:virtual bool
  handleDecision( ITestDecisionFn,
                  const IString& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method executes the decision function and returns the result.

You can override this function to customize the execution of the decision function, for example, to write information to log or to catch certain exceptions thrown by the function. To override handleDecision, add code before and/or after calling the decision functions. If not overridden, handleDecision just calls the decision function and returns the result boolean value.

ITestDecisionFn
The decision function to run.

IString&
The key of the decision function. If not overridden, the key is ignored.

loadDecisions

```
protected:virtual void loadDecisions()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework method for subclasses to override.

Inside loadDecisions subclasses must first call the inherited implementation (IDervied::loadDecisions must call IBase::loadDecisions). Then loadDecisions should register each key-decision function pair by calling addDecisions(theKey, theFunc), where theKey and theFunc are the key and the address of the decision member function. If a subclass has overridden a decision member function in its base class, it must register the keyword and the subclass's version of the function in loadDecisions. Do not call loadDecisions; it is called by ITestMultiplexer.

print

```
protected:virtual void print(ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Prints the keys used for running the test to the given ITieredTextBuffer. Do not call this function directly.

test

```
protected:virtual void test()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Runs specified decision functions and returns logical AND of the results.

Runs whatever decision functions have previously been specified by their keys and calls setSuccess with the logical AND of the results.

Note: Contrary to usual ITest practice, subclasses should not override this method. Instead, subclasses should do the following:

- Declare a new decision function for each test.
- Define a key for each decision function.
- Register each key-decision function pair in an overridden version of the loadDecisions methods.

Do not call this method directly; it is called by ITest::run.

Decision Key Methods

Class member functions used to indicate which decision test functions should be executed.

decisionKey

```
public:virtual void decisionKey(IString& dest) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the argument dest to the current decision key. If input AsKey is set to zero, returns a NULL string.

inputsAsKeys

```
public:virtual unsigned int inputsAsKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current value of fInputsAsKeys.

setDecisionKey

```
public:virtual bool setDecisionKey(const IString& key)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the key that will be used in subsequent calls to test. Returns false and does nothing if the key is not valid; otherwise, returns true.

Use the key ITestMultiplexer::kAllDecisions to specify that all decision functions are to be run. The setDecisionKey function calls setInputAsKeys(0) so that no inputs will be interpreted as keys.

setInputsAsKeys

```
public:virtual void setInputsAsKeys(unsigned int howMany)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows you to specify how many of the ITest input arguments, starting with the first one, will be interpreted as keys.

Calling setInputsAsKeys with a nonzero value overrides any current key set via setDecisionKey, and causes decisionKey to return an empty string. Calling setInputsAsKey(0) will reactivate any key previously set via setDecisionKey, or the key kAllDecisions if setDecisionKey has not been called.

kAllDecisions

```
public:static const IString & kAllDecisions
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The key ITestMultiplexer::kAllDecisions has the effect of running all decision functions and doing a logical AND of the test results. Initial value is ″All″.

ITestMultiplexer - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInfo	815
copyInputs	811	exceptionStage	812
lookupInfo	816	operator =	812
outputTextStream	816	run	812
runCount	817	setInputs	811
success	813		

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	cleanup	814

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
isReset	817	isSuccessUndecided	818
ITest	812	reset	814
setStickySuccess	818	setSuccess	818
setup	814	test	815

Inherited Protected Data

None

ITieredTextBuffer

Clients use ITieredTextBuffer as if it were a C++ ostream object. It has operator<< methods for all built-in types.

To write out diagonostic text from a test, use the member function outputTextStream, which recognizes the standard C++ << operator for all built-in types. Text output produced with outputTextStream displays to the console and is also saved in a ITieredTextBuffer within the test. You can log the test, including the text buffer. If a test fails, you can retieve the associated diagnostic text to determine the cause of the failure.

outputTestStream returns a pointer to a ITieredTextBuffer object that contains the text. Objects of the class ITieredTextBuffer act like C++ ostream objects. They support << operators for all built-in types.

ITieredTextBuffer has some additional functionality beyond that of ostream to support retrieving textual information for later analysis. Information is marked with a level of importance. This permits showing a little or a lot of detail when the text is printed.

Most text occupies the tier kNormal. Important information that should not be missed occupies tier kGeneral or kHeadline. Detailed information that can usually be ignored occupies tier kDetail or kDebug.

This is a list of available tiers in order of least detailed to most detailed:

```
kTop: Most general information
kHeadline: One-line information
kGeneral: General interest
kNormal: Default
kDetail: Detail not usually needed
kDebug: Only needed during debugging
kBottom: Lowest tier, most detailed
```

Class header file: ittxtbuf.hpp

ITieredTextBuffer - Hierarchy List

ITieredTextBuffer

ITieredTextBuffer - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITieredTextBuffer

```
public:virtual ~ITieredTextBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor.

ITieredTextBuffer

Overload 1

```
public:ITieredTextBuffer(const ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Overload 2

```
public:ITieredTextBuffer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Assignment Operator

Standard assignment operator.

operator =

```
public:ITieredTextBuffer& operator =(const ITieredTextBuffer&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment operator.

Echo and Output Configuration

Member functions used to control the output buffer appearance and configuration.

echoTier

```
public:ETier echoTier() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ETier level at which echo is set.

indent

```
public:unsigned short indent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current indentation level setting.

openLog

```
public:void openLog(const char* logFileName)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to open the specified log file. Causes output to be displayed to standard output as well as to the log file.

logFileName
String value of the log file to open.

popTier

```
public:ETier popTier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Restore tier to value before last pushTier.

Pops the tier at the top of the stack, as previously pushed on by pushTier, and returns it. The tier is set to the value it had before that last call to pushTier.

pushTier

```
public:void pushTier(ETier newTier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes tier to new setting and saves old value.

Pushes newTier onto an internal stack of ETiers and makes it the new tier by calling setTier. A subsequent call to popTier will restore the previous tier. If the internal stack is full, the stack is unchanged and newTier does not get pushed.

setEchoTier

```
public:void setEchoTier(ETier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the tier above which text is echoed to console.

This method sets the tier level at which echoing occurs. ITieredTextBuffer objects echo all output sent to them (via the operator<< method) to standard output. Echoing only happens for text above a certain tier, which is set by these methods. Subclasses may override setEchoTier if they want notification when the echo tier changes.

setIndent

```
public:unsigned short setIndent(unsigned short)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the indent level for subsequent output via the operator<< methods. The short parameter specifies the level of indentation. If new indent is zero, text will be flushed left. Indentation only affects text that is added to this object via one of the << operators.

setRelativeIndent

```
public:unsigned short setRelativeIndent(short)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the indent level for subsequent output relative to the current indent level. The short parameter specifies the level of indentation.

setTier

```
public:ETier setTier(ITieredTextBuffer::ETier newTier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes the tier of all subsequently streamed text to newTier. It only affects future calls to operator<<. setTier pins newTier to the range kTop...kBottom.

tier

```
public:ETier tier() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current tier object.

Standard C++ << Operators

Standard output operators for use with ITieredTextBuffers.

operator <<

C++ ostream-style output for use with ITieredTextBuffer.

Overload 1

```
public:virtual ITieredTextBuffer& operator <<(const char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for char*.

Overload 2

```
public:virtual ITieredTextBuffer& operator <<(const long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for long.

Overload 3

```
public:virtual ITieredTextBuffer& operator <<(const double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for double.

Overload 4

```
public:virtual ITieredTextBuffer& operator <<(const unsigned char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for unsigned char.

Overload 5

```
public:virtual ITieredTextBuffer& operator <<(const char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for char.

Overload 6

```
public:virtual ITieredTextBuffer& operator <<(const unsigned int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for unsigned int.

Overload 7

```
public:virtual ITieredTextBuffer&
  operator <<( const unsigned short )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for unsigned short.

Overload 8

```
public:virtual ITieredTextBuffer& operator <<(const unsigned long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for unsigned long.

Overload 9

```
public:virtual ITieredTextBuffer& operator <<(const signed char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for signed char.

Overload 10

```
public:virtual ITieredTextBuffer& operator <<(const short)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for short.

Overload 11

```
public:virtual ITieredTextBuffer&
  operator <<( const unsigned long long )
```

This is supported on **AIX OS/400 z/OS**

Note: This operator is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 12

```
public:virtual ITieredTextBuffer& operator <<(const int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for int.

Overload 13

```
public:virtual ITieredTextBuffer& operator <<(const long long)
```

This is supported on **AIX OS/400 z/OS**

Note: This operator is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 14

```
public:virtual ITieredTextBuffer& operator <<(const long double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for long double.

Overload 15

```
public:virtual ITieredTextBuffer& operator <<(const float)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator for float.

Overload 16

```
public:virtual ITieredTextBuffer& operator <<(endlfn)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

C++ ostream-style output operator that allows users to call endl.

ITieredTextBuffer - Enumerations

ETier

kTop Top tier, most general information

kHeadline
One-liner information

kGeneral
Information of general interest

kEphemeral
Only shows on screen; not saved

kNormal
Default tier

kNormal
Default tier

kDetail
Detailed information not usually needed

kDebug
Information only needed during debugging

kBottom
Lowest tier, most detailed information

ITieredTextBuffer - Type Definitions

endlfn

```
typedef ITieredTextBuffer & ( * endlfn)( ITieredTextBuffer & )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Allows you to call endl with ITieredTextBuffer.

ITieredTextBuffer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITimingTest

An abstract base class that represents the tests that measure the median and time required to execute a well-defined operation. It is the main class used to measure CPU performance.

Instead of the usual setup, test, and cleanup framework methods, ITimingTest has timingSetup, timingTest, and timingCleanup framework methods. These methods behave somewhat differently from their namesakes. Subclasses place the code to be timed in timingTest. The methods timingSetup and timingCleanup allow subclasses to do any setup and cleanup for the timingTest method; code in these methods is NOT timed.

The ITimingTest base class enables you to write tests that measure the time taken by a specific operation to complete. The derived class IStartStopTimingTest allows you more control in timing events.

ITimingTest expects the operation to show some variation in execution time due to multitasking, varying CPU load, and other variable conditions. Consequently, the class takes multiple measurements and reports the median.

ITimingTest measures the time needed to execute the timingTest member function. It does this by taking some number of samples, the ″sample count″. Each sample consists of a call to timingSetup, some number of calls to timingTest, and a call to timingCleanup.

The calls to timingTest happen in a tight loop, the ″timing loop″. Derived classes, or the caller through a RunTest option, can control how many iterations comprise the timing loop by changing the ″timing count″. The total time taken by the timing loop is divided by the number of iterations to get a single ″sample time″. The median of all sample times is taken as the estimate of the time to execute the timingTest member function.

To create a timed test, do the following:

1. Derive your test class from ITimingTest.
2. Override timingSetup, if necessary.
3. Override timingTest.
4. Override timingCleanup, if necessary.

The default settings are as follows:

```
kDefaultTimingCount = 500,
kDefaultSampleCount = 10,
kDefaultMaximumSampleWindows = 10
kDefaultAutomaticSamplingTolerance = 0.1
kDefaultMinimumCorrelation = .4
```

Class header file: itimetst.hpp

ITimingTest - Hierarchy List

ITest

ITimingTest

IStartStopTimingTest

ITimingTest - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITimingTest

```
public:virtual ~ITimingTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor.

ITimingTest

Overload 1

Standard copy constructor.

```
public:ITimingTest(const ITimingTest& source)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy Constructor.
source ITimingTest to be copied.

Overload 2

```
protected:ITimingTest( ioc::unumeric_t timingCount,
            ioc::unumeric_t sampleCount )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

Default constructor.

```
protected:ITimingTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This default constructor sets the timing count and sample count to default values, with no maximum allowed median time.

Constructs an ITimingTest specifying a timing count, sample count, maximum sample count, minimum correlation, and an automatic sample tolerance. The default values are:

```
kDefaultTimingCount = 500,
kDefaultSampleCount = 10,
kDefaultMaximumSampleWindows = 10
kDefaultAutomaticSamplingTolerance = 0.1
kDefaultMinimumCorrelation = .4
automatic sampling ON
no maximum allowed median time
```

Advanced Subclass Specialization Methods

Advanced timing member functions. These should only be used by those users with a good understanding of the timing tests.

doUntimedTest

```
protected:virtual void doUntimedTest()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Run the timingTest without performing the timing. (e.g., execute only timingSetup, timingTest, and timingCleanup). Useful to debug the timingTest without concern for the timing aspects.

oneEmptySample

```
protected:virtual double oneEmptySample()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Runs an empty sample. Useful to determine the amount of time necessary for the overhead of running timingTest, the function call to an empty test function (e.g., timingTest return 0.0), and timingCleanup.

Return

The time to run the empty sample.

oneSample

```
protected:virtual double oneSample()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determine the time it takes to run one sample of the timingTest.

Return

The time to run the one sample.

samplingErrorMicroseconds

```
protected:virtual double samplingErrorMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the sampling error.

Assignment Operator

Use this operator to assign an ITimingTest class to another.

operator =

```
public:ITimingTest& operator =(const ITimingTest& rhs)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator.

rhs ITimingTest object used for assignment values.

Automatic Sampling Methods

Member functions used to configure the automatic timing sampling.

automaticSampleTolerance

```
public:double automaticSampleTolerance() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the threshold value for automatic sampling.

automaticSampling

```
public:bool automaticSampling() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if automatic sampling is done, false otherwise.

minimumSampleCorrelation

```
public:double minimumSampleCorrelation() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the minimum acceptable correlation between samples.

setAutomaticSampleTolerance

```
public:void setAutomaticSampleTolerance(double tolerance)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the threshold value for automatic sampling. Samples will continue to be taken until the error on the measured mean is below this threshold value. For example, if the Tolerance is 0.05, then samples will continue to be taken until the error on the mean is less than or equal to 5 percent.

tolerance
The threshold value for automatic sampling

setAutomaticSampling

```
public:void setAutomaticSampling(bool doAutomaticSampling)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Controls whether automatic sampling is done.

doAutomaticSampling
If true, samples are taken until the error on the measured mean reaches a desired percentage, the tolerance. This allows the operation being timed to ″settle″ if the first measurements being taken are too high or low for some reason. Samples taken prior to settling are discarded. If false, then the first sampleCount samples are always taken and accepted.

setMaximumSampleWindows

```
public:void setMaximumSampleWindows(ioc::unumeric_t maxWindows)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the maximum number of sample windows that are taken.

maxWindows
Number of sample windows to take

setMinimumSampleCorrelation

```
public:void setMinimumSampleCorrelation(double minSampleCorr)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the minimum acceptable correlation between samples. If the correlation between samples exceeds this number, a message will be printed and the actual correlation will be posted.

minSampleCorr
The minimum correlation: 0 < minSampleCorr < 1

kDefaultAutomaticSamplingTolerance

```
public:static const double kDefaultAutomaticSamplingTolerance
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default automatic sampling tolerance. Initial value set to 0.1.

kDefaultMinimumCorrelation

```
public:static const double kDefaultMinimumCorrelation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default minimum correlation. Initial value set to 0.4.

Timing Methods

Class member functions used to control and perform the timing tests.

calibrate

```
public:void calibrate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Measures the empty method time.

calibratedMeanMicroseconds

```
public:double calibratedMeanMicroseconds()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the mean of all sampled times.

calibratedMedianMicroseconds

```
public:double calibratedMedianMicroseconds()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the estimated median time for the timingTest method to execute minus the empty method time, in microseconds.

clockNowOverhead

```
public:double clockNowOverhead() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the overhead required to read the clock. This is a measured value used to compute other error tolerances.

continualCalibration

```
public:bool continualCalibration() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if calibrate is called for each sample; false otherwise.

copyInfo

```
public:virtual void copyInfo()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds information about input syntax and a timing test description. Can be overridden to provide metainformation.

copySamples

```
public:void copySamples(ioc::numeric_t& count, double *&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method sets a pointer to point at an ordered array of each sample value.

Note: The caller owns its storage for the elements in the array.

count The number of elements of the array pointed to

double *&
The array pointer to set.

correlationMicroseconds

```
public:double correlationMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the correlation between samples.

correlationTooLarge

```
public:bool correlationTooLarge() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the correlation between samples exceeds the minimum correlation specified by the user (using setMinimumSampleCorrelation).

errorMicroseconds

```
public:double errorMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the 95% confidence interval for the test.

errorTooLarge

```
public:bool errorTooLarge() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the relative error from relativeErrorMicroseconds is greater than the tolerance specified by the tester on the command line or by calling setAutomaticSampleTolerance.

maximumAllowedMedianMicroseconds

```
public:double maximumAllowedMedianMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the time limit for this timing test.

maximumSampleWindows

```
public:ioc::unumeric_t maximumSampleWindows() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum number of sample windows.

relativeErrorMicroseconds

```
public:double relativeErrorMicroseconds() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ratio of the 95% confidence interval for the mean (as returned by errorMicroseconds()) to the mean itself.

sampleCount

```
public:ioc::unumeric_t sampleCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of samples that will be taken for timing and calibration.

samplesDiscarded

```
public:ioc::unumeric_t samplesDiscarded() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of samples discarded in the case that automatic timing is on.

samplesTaken

```
public:ioc::unumeric_t samplesTaken() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of samples actually taken for a full test.

setContinualCalibration

```
public:void setContinualCalibration(bool continuallyCalibrate)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Controls whether calibrate is called for each sample.

Continual calibration is off by default under the assumption that the empty method time is relatively constant. If the client or subclass is varying the load, then continual calibration may provide more accurate timing results.

continuallyCalibrate

bool; If true, timing is set to be continually calibrated

setMaximumAllowedMedianMicroseconds

```
public:void setMaximumAllowedMedianMicroseconds(double timeLimit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the time limit for this timing test. If the final calibrated median time exceeds the specified time limit, then the test is considered a failure and setSuccess(false) is called. If setMaximumAllowedMedianMicroseconds is never called (the default case) then no time limit will be set, and all runs will be reported to be passing tests.

timeLimit

Maximum allowed time

setSampleCount

```
public:void setSampleCount(ioc::unumeric_t sampleCount)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the number of samples that will be taken for timing and calibration. This is also the number of samples that will be taken (in a moving window) for automatic timing.

sampleCount

The number of samples to take

setTimingCount

```
public:void setTimingCount(ioc::unumeric_t timingCount)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the number of times the timingTest method will be called in a loop during a single sample.

Note: This timing count also applies to the calibration method.

timingCount
The number of calls to make within a single sample

setTimingOn

```
public:void setTimingOn(bool timingOn)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Controls whether timing operations are done for the timingTest method.

timingOn
A bool. If true, multiple samples are taken, each sample consisting of a call to timingSetup, many calls to timingTest, and a call to timingCleanup. If false, each run makes a single call to timingSetup, timingTest, and timingCleanup. In addition, echoing to the console is suppressed.

timingCount

```
public:ioc::unumeric_t timingCount() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of times timingTest will be called in a loop during a single sample.

timingOn

```
public:bool timingOn() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if timing operations are done for the timingTest. Otherwise returns false.

cleanup

```
protected:virtual void cleanup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a framework cleanup method. It is called once after the test is run and should contain whatever code is necessary to cleanup after setup. It also must call the parent ITimingTest::cleanup method.

print

```
protected:virtual void print(ITieredTextBuffer& out)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Prints information about the object to the given tiered text buffer.

out The buffer where information about the test is printed.

setup

```
protected:virtual void setup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework setup method. It is called once before the test is run.

The test should override this function to execute any setup code that only needs to be called once.

It is also a good place to parse command line arguments. If this method is overridden it must call the parent ITimingTest::setup method.

test

```
protected:virtual void test()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework test method. ITimingTest overrides the ITest::test method; subtests of ITimingTest should not override it. ITimingTest::test first calls calibrate if this has not been done before. It then takes sampleCount timing samples if automatic sampling is off, or automatically takes enough samples to achieve the desired tolerance, as returned by automaticSamplingTolerance, up to a maximym of maximymSampleCount samples.

Note: Do NOT override this method.

timingCleanup

```
protected:virtual void timingCleanup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Performs any required cleanup for n consecutive calls to timingTest, where n is the timing count. Called by the framework once per sample. Subclasses must override timingCleanup if any cleanup is necessary after calls to timingTest.

timingSetup

```
protected:virtual void timingSetup()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Performs any required setup for n consecutive calls to timingTest, where n is the timing count. Called by the framework once per sample. Subclasses must override timingSetup if any setup is necessary before calls to timingTest.

timingTest

```
protected:virtual void timingTest() = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A framework method for timing tests. Subclasses must override timingTest and place the code to be timed in the overridden method. If this code requires any setup and/or cleanup, subclasses should override timingSetup and timingCleanup and place the appropriate code in each.

For each timing sample, ITimingTest calls timingSetup once, followed by timingCount calls to timingTest, followed by a call to timingCleanup. The loop of calls to timingTest is timed, and this time is divided by timingCount to get the uncalibrated sample time. The median empty time is subtracted from this to get the calibrated sample time.

ITimingTest - Enumerations

Default constants used by the timing tests.

ITimingTest - Inherited Member Functions and Data

Inherited Public Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
~ITest	812	copyInputs	811
exceptionStage	812	lookupInfo	816
operator =	812	outputTextStream	816
run	812	runCount	817
setInputs	811	success	813

Inherited Public Data

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
kComponentKey	816	kDescriptionKey	817
kInputSyntaxKey	817	kTargetClassKey	817
kTestNameKey	817	kTestTypeKey	817
kTestVersionKey	817		

Inherited Protected Functions

ITest			
Definition	**Page Number**	**Definition**	**Page Number**
addInfo	813	isReset	817
isSuccessUndecided	818	ITest	812
print	814	reset	814
setStickySuccess	818	setSuccess	818
test	815		

Inherited Protected Data

None

ITimingTestStopwatch

ITimingTestStopwatch acts as a client-controllable stopwatch for the IStartStopTimingTest class.

Class header file: issttest.hpp

ITimingTestStopwatch - Hierarchy List

ITimingTestStopwatch

ITimingTestStopwatch - Member Functions and Data by Group

Constructors & Destructor

Class constructors and destructors.

~ITimingTestStopwatch

```
public:~ITimingTestStopwatch()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs objects of this class.

ITimingTestStopwatch

Create a client-controllable stopwatch for the IStartStopTimingTest class.

Overload 1

```
public:ITimingTestStopwatch()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a new stopped and reset stopwatch.

Overload 2

```
public:ITimingTestStopwatch(const ITimingTestStopwatch&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a new ITimingTestStopwatch object from an existing object.

Assignment Operators

Class standard assignment operators.

operator =

```
public:ITimingTestStopwatch&
  operator =( const ITimingTestStopwatch& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Standard assignment operator.

Stopwatch Functions

start

```
public:void start()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts a stopwatch without resetting it, so that time can be accumulated into the stopwatch.

stop

```
public:void stop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops time measurements but does not reset the stopwatch.

ITimingTestStopwatch - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITrace

The ITrace class provides module tracing within the Open Class Library. Whenever an exception is thrown by the library, trace records are output with information about the exception. You can use the **ICLUI_TRACE** and **ICLUI_TRACETO** environment variables to redirect the trace output to a file. The output trace records contain the following:

- Error message text
- Error ID
- Class name
- Information from the class IExceptionLocation

By default, the library disables tracing. You can set tracing on by entering **ICLUI_TRACE=ON** in the environment.

Also, by default, the library attaches a prefix to the trace entry containing a sequence number followed by the process and thread where the trace call occurred. You can remove prefix area tracing by entering **ICLUI_TRACE=NOPREFIX** in the environment. Doing so has the side effect of turning tracing on.

You can set the output location of tracing by entering one of the following in the environment:

- **ICLUI_TRACETO=STDERR** for the standard error stream (stderr)
- **ICLUI_TRACETO=STDOUT** for the standard output (stdout)
- **ICLUI_TRACETO=FILE** for a file stream specified by the environment ICLUI_TRACEFILE
- **ICLUI_TRACETO=QUEUE** for a queue

Specifying any of the preceding locations has the side effect of turning tracing on.

In addition to turning the trace options on and off in the environment, the library also provides static member functions to do the same thing under program control.

Tracing can be set on and off using the static functions enableTrace() and disableTrace().

The output location of tracing can be changed to stderr or stdout by using the static functions writeToStandardError() or writeToStandardOutput().

Prefix area tracing can be enabled or disabled by using the static function enableWritePrefix(), or disableWritePrefix().

The library supports trace input as IStrings or character arrays, and the library automatically adds a line feed on all trace calls.

To enable you to compile the trace calls in and out of your code, the Open Class Library provides the following sets of macros for tracing modules and data:

- The library defines IC_TRACE_RUNTIME by default. The following macros are expanded:

  ```
  IMODTRACE_RUNTIME()  IFUNCTRACE_RUNTIME()  ITRACE_RUNTIME()
  ```

- If you define IC_TRACE_DEVELOP, the following macros, in addition to the RUNTIME macros, are expanded:

  ```
  IMODTRACE_DEVELOP()  IFUNCTRACE_DEVELOP()  ITRACE_DEVELOP()
  ```

- If you define IC_TRACE_ALL, the following macros, in addition to the RUNTIME and DEVELOP macros, are expanded:

  ```
  IMODTRACE_ALL()      IFUNCTRACE_ALL()      ITRACE_ALL()
  ```

The IMODTRACE version of the macros accepts as input a module name that it uses for construction and destruction tracing.

The IFUNCTRACE version of the macros accepts no input and uses the predefined identifier __FUNCTION__ for construction and destruction tracing.

The ITRACE version of the macros accepts a text string to be written out.

AIX Considerations

The default output location of tracing is standardOutput. Setting the output location of tracing to queue has the same effect in X-Motif as setting it to standardOutput.

OS/2 Considerations

In OS/2, the library supports the environment variables ICLUI TRACE and ICLUI TRACETO, in addition to ICLUI_TRACE and ICLUI_TRACETO.

The default output location of tracing is the OS/2 queue \\QUEUES\\PRINTF32. You can display this queue using the program PMPRTF32.EXE.

z/OS Considerations

The default output location of tracing is standardOutput. Setting the output location of tracing to queue has the same effect as setting it to standardOutput.

Class header file: itrace.hpp

ITrace - Hierarchy List

ITrace

ITrace - Member Functions and Data by Group

Constructors & Destructor

You can construct and destruct objects of this class by using the default constructor and the provided destructor.

~ITrace

```
public:~ITrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITrace

Construct an ITrace object using one of these constructors.

Overload 1

```
public:ITrace( const ioc::unichar_t* traceName,
        ioc::numeric_t lineNumber = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ITrace( const char* traceName,
        ioc::numeric_t lineNumber = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:ITrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor creates an ITrace object, but no logging occurs on construction or destruction.

Debug Support

Debug support methods.

writeDebugLocation

Writes the passed debug location information out.

Overload 1

```
public:static void
  writeDebugLocation( const ioc::unichar_t* str,
                      const IExceptionLocation& location )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:static void
  writeDebugLocation( const char* str,
                      const IExceptionLocation& location )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enabling and Disabling

Use these members to enable or disable tracing, as well as to query whether tracing is on.

disableTrace

```
public:static void disableTrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disables trace entries from being written.

enableTrace

```
public:static void enableTrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enables trace entries to be written.

isTraceEnabled

```
public:static bool isTraceEnabled()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether tracing is currently enabled.

Format

Use these members to enable, disable, and query the formatting options for writing trace output.

disableUnicodeTrace

```
public:static void disableUnicodeTrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disable Unicode trace output. The default is disabled.

Unicode trace output is only supported with trace file. (i.e. ICLUI_TRACETO=file)

disableWriteLineNumber

```
public:static void disableWriteLineNumber()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disables the tracing of line number information.

disableWritePrefix

```
public:static void disableWritePrefix()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disables the writing of the process ID, the thread ID, and the output line number to trace.

enableUnicodeTrace

```
public:static void enableUnicodeTrace()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enable Unicode trace output. The default is disabled.

Unicode trace output is only supported with trace file. (i.e. ICLUI_TRACETO=file)

enableWriteLineNumber

```
public:static void enableWriteLineNumber()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enables the tracing of line number information.

enableWritePrefix

```
public:static void enableWritePrefix()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Enables the writing of the process ID, the thread ID, and the output line number to trace.

Portability Considerations

- On z/OS C/C++, both processId and threadId are always set to 1.
- On AIX implementations, threadId is always set to 1. processId is the id of the process.

- On OS/400, processId is always set to 0 and threadId is always set to 1.

isWriteLineNumberEnabled

```
public:static bool isWriteLineNumberEnabled()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether line numbers are currently being written.

isWritePrefixEnabled

```
public:static bool isWritePrefixEnabled()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines whether the line count prefix is being written.

setMarker

```
public:static void
  setMarker( const char* begin_marker = IOCMSG ( ioc::kADD ),
             const char* end_marker = IOCMSG ( ioc::kHYPHEN ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Set the default marker to use for trace output.

begin_marker
Marks the entering of a function call

end_marker
Marks the leaving of a function call

For example, you can set the markers to braces so that you can easily identify the pair in an editor:

```
...
ITrace::setMarker("{ ", "} // ");
...
```

Output:

```
{ MyFunction()
  ...
} // MyFunction()
```

setMarkerW

The Unicode version of setMarker().

Overload 1

```
public:static void
  setMarkerW( const ioc::unichar_t* begin_marker =
              ( const ioc::unichar_t * ) WIOCMSG ( ioc::kWADD ),
              const ioc::unichar_t* end_marker =
              ( const ioc::unichar_t * ) WIOCMSG ( ioc::kWHYPHEN ) )
```

This is supported on **WIN OS/2 AIX OS/400**

Overload 2

```
public:static void
  setMarkerW( const ioc::unichar_t* begin_marker = ( const ioc::unichar_t * ) wIoc
              const ioc::unichar_t* end_marker = ( const ioc::unichar_t * ) wIocMs
```

This is supported on **z/OS**

Output Operations

Use these members to do the following:

- Write trace data to the current trace location
- Query the current trace location
- Set the current trace location

traceDestination

```
public:static ITrace::Destination traceDestination()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the trace output destination for this trace object. The returned value is an enumerator provided by ITrace::Destination.

write

Writes the specified text.

Overload 1

```
public:static void write(const char* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

text The text to write as a character string.

Overload 2

```
public:static void write(const ioc::unichar_t* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:static void write(const IString& text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

text The text to write as an IString.

writeToFile

```
public:static void writeToFile()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Set the location for output to a file. Using this function is equivalent to setting the environment variable **ICLUI_TRACETO=FILE**.

In order for this to work another environment variable ICLUI_TRACEFILE needs to be set.

writeToQueue

```
public:static void writeToQueue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the location for output to \\QUEUES\\PRINTF32 in the OS/2 environment.

Sets the location for output to \\.\MAILSLOT\PRINTF32 in the Win32 environment.

Sets the location for output to Job log in the OS/400 environment.

AIX Considerations

In the AIX environment, this member function is equivalent to writeToStandardOutput.

z/OS Considerations

In the z/OS environment, this member function is equivalent to writeToStandardOutput.

writeToStandardError

```
public:static void writeToStandardError()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the location for output to the standard error stream.

writeToStandardOutput

```
public:static void writeToStandardOutput()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the location for output to the standard output stream. Using this function is equivalent to setting the environment variable **ICLUI_TRACETO=OUT**.

Note: STDOUT is a synonym for OUT.

writeFormattedString

Writes the trace data after formatting, which includes the following:

- Adding the prefix, if necessary
- Updating any new lines embedded in the string to include the prefix

string Any trace information you want to write.

marker

When the Open Class Library uses this function, it specifies a character string to mark, or distinguish, whether the trace statement is entering or leaving a function. You can specify **marker** for any purpose.

Overload 1

```
protected:static void
  writeFormattedString( const ioc::unichar_t* string,
                        const ioc::unichar_t* marker )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static void
  writeFormattedString( const char* string,
                        const char* marker )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:static void
  writeFormattedString( const IString& string,
                        const char* marker )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

writeString

Writes to the output device without formatting.

text Any trace information you want to write.

Overload 1

```
protected:static void writeString(const char* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static void writeString(const ioc::unichar_t* text)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Thread ID

Use these members to query the thread ID.

threadId

Returns the current thread identifier.

Overload 1

```
protected:static unsigned long threadId()
```

This is supported on **WIN OS/2 AIX z/OS**

Portability Considerations

In environments that do not support kernel threads, this function always returns a 1.

Overload 2

```
protected:static unsigned long long threadId()
```

This is supported on **OS/400**

Portability Considerations

In environments that do not support kernel threads, this function always returns a 1.

ITrace - Enumerations

Destination

These enumerators specify the destination of the trace data:

queue Sends the trace data to the queue.

standardError
Sends the trace data to the standard error stream (stderr).

standardOutput
Sends the trace data to the standard output (stdout).

file Sends the trace data to a file specified by environment variable ICLUI_TRACEFILE.

When used on the following platforms, the queue enumerator is not supported, and queue tracing goes to stdout:

- AIX
- z/OS
- OS/400

ITrace - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

ITrace			
Definition	**Page Number**	**Definition**	**Page Number**
`pid`	846		

Inherited Protected Data

None

Chapter 6. Collection Classes

IABag

A **bag** is an unordered collection of zero or more elements with no key. Multiple elements are supported. A request to add an element that already exists is not ignored.

An example of using a bag is a program for entering observations on species of plants and animals found in a river. Each time you spot a plant or animal in the river, you enter the name of the species into the collection. If you spot a species twice during an observation period, the species is added twice, because a bag supports multiple elements. You can locate the name of a species that you have observed, and you can determine the number of observations of that species, but you cannot sort the collection by species, because a bag is an unordered collection. If you want to sort the elements of a bag, use a sorted bag instead.

The following rule applies for duplicates: If bag P contains the element X **m** times and bag Q contains the element X **n** times, then the **union** of P and Q contains the element X **m+n** times, the **intersection** of P and Q contains the element X **MIN(m,n)** times, and the **difference** of P and Q contains the element X **m-n** times if **m** is > **n**, and **zero** times if **m** is <= **n**.

Class header file: iabag.h

IABag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - **IABag**
        - IGBagAsList
        - IGBagAsHshTable
        - IGBagAsAvlTree
        - IGBagAsDilTable
        - IGBagAsBstTree
        - IGBag
        - IGBagAsTable
        - IRBag

IABag - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IABag

```
public:~IABag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

© Copyright IBM Corp. 2001

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IABag

Constructs an IABag object.

Overload 1

```
public:IABag(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IABag(IABag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IABag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the

required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Miscellaneous

Use these member functions to perform miscellaneous operations.

addDifference

```
public:virtual void
  addDifference( IABag < Element > const&,
                 IABag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IABag::differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map

addIntersection

```
public:virtual void
  addIntersection( IABag < Element > const&,
                   IABag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IABag::intersectionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map

addUnion

```
public:virtual void
  addUnion( IABag < Element > const&,
            IABag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IABag::unionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded

• IKeyAlreadyExistsException, if the collection is a map or a sorted map

differenceWith

```
public:virtual void differenceWith(IABag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void intersectionWith(IABag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

numberOfDifferentElements

```
public:INumber numberOfDifferentElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different elements in the collection.

operator !=

```
public:bool operator !=(IABag < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see IABag::operator ==.

operator ==

```
public:bool operator ==(IABag < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Meaning
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

setToNextDifferentElement

```
public:bool setToNextDifferentElement(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception

- ICursorInvalidException

unionWith

```
public:virtual void unionWith(IABag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map

IABag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IACollection

IACollection is an abstract class. It cannot be used to create any objects. This class defines the base interfaces for the collection classes.

The following abstract classes are derived from IACollection:
- IAEqualityCollection
- IAKeyCollection
- IAOrderedCollection

The concrete class IHeap is defined by IACollection.

Class header file: iacllct.h

IACollection - Hierarchy List

IMetaCollection

IACollectionBase

IACollection

IAOrderedCollection
IAEqualityCollection
IAKeyCollection
IAHeap

IACollection - Member Functions and Data by Group

Constructors & Destructor

IACollection is an abstract class, therefore you cannot create or destroy any objects of this class.

~IACollection

```
public:virtual ~IACollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IACollection

Constructs an IACollection object.

Overload 1

```
public:IACollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IACollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For

some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 3

```
protected:IACollection(IACollection < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Collection Status

Member functions that test the status of the collection.

isBounded

```
public:bool isBounded() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is bounded.

isConsistent

```
public:bool isConsistent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

For internal use only.

isEmpty

```
public:bool isEmpty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is empty.

isFull

```
public:bool isFull() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is bounded and contains the maximum number of elements; that is, if (numberOfElements() == maxNumberOfElements()).

maxNumberOfElements

```
public:INumber maxNumberOfElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum number of elements the collection can contain.

Precondition

The collection is bounded.

Exceptions
- INotBoundedException

numberOfElements

```
public:INumber numberOfElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements the collection contains.

Cursor Support

Member functions used to access, add or remove elements using cursors.

add

```
public:virtual bool add(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the collection is unique (with respect to elements of keys) and the element of key is already contained in the collection, sets the cursor to the existing element in the collection without adding the element. Otherwise, it adds the element to the collection and sets the cursor to the added element. In sequential collections, the given element is added as the last element. In sorted collections, the element is added at a position determined by the element or key value. Adding an element will either use the element's copy constructor or the assignment operator provided for the element type, depending on the implementation variant you choose.

Preconditions
- The cursor must belong to the collection.
- If the number is bounded and unique, the element of key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElement() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If an element as added, all cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Return Value

Returns true if the element was added.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

elementAt

Returns a reference to the element pointed to by the given cursor.

Precondition

The cursor must belong to the collection and must point to an element of the collection.

Exception
- ICursorInvalidException

Overload 1

```
public:Element const& elementAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Note: For this version that has the **const** suffix, do not manipulate the element of the key of the element in the collection in a way that changes the positioning property of the element.

Overload 2

```
public:Element& elementAt(ICursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

newCursor

```
public:virtual ICursor* newCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a cursor. The cursor is initially invalid. Returns a pointer to the cursor.

Exceptions
- IOutOfMemory

newMngCursor

```
public:ICursor* newMngCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a managed cursor. The cursor is initially invalid. Returns a pointer to the cursor.

Exceptions
- IOutOfMemory

removeAt

```
public:virtual void removeAt(ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the element pointed to by the given cursor. The given cursor is invalidated.

Note: It is undefined whether the destructor for the removed element is called or whether the element will only be destructed with the collection destructor. For example, in a tabular implentation, a destructor will only be called when the whole collection is destructed, not when a single element is removed.

Precondition

The cursor must belong to the collection and must point to an element of the collection.

Side Effects

- All cursors of this collection, except the given cursor, become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception

- ICursorInvalidException

replaceAt

```
public:virtual void replaceAt(ICursor const&, Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the element pointed to by the cursor with the given element.

Preconditions

- The cursor must belong to the collection and must point to an element of the collection.
- The given element must have the same positioning property as the replaced element.

Side Effects

- Collection classes supporting notifications send a replaceId notification.

Exception

- ICursorInvalidException
- IInvalidReplacementException

setToFirst

```
public:bool setToFirst(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first element of the collection in iteration order. If the collection is empty (if no first element exists), it invalidates the given cursor.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if the collection is not empty.

Exception

- ICursorInvalidException

setToNext

```
public:bool setToNext(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element of the collection in iteration order. If no more elements are left to be visited, the given cursor will no longer be valid.

Precondition

The cursor must belong to the collection and must point to an element.

Return Value

Returns true if there is a next element.

Exception
- ICursorInvalidException

Do For All Elements

Member functions that apply actions to all the elements of the collection.

allElementsDo

Overload 1

```
public:bool
  allElementsDo( bool ( * applicatorFunction ) ( Element&,
                 void * ) , void* additionalArgument = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements in the collection until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Notes:

1. The given function must not remove elements from or add elements to the collection. If you want to remove elements, you can use the removeAll() function with a property argument.
2. For the non-const version of allElementsDo(), the given function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the given function returns true for every element it is applied to. Returns false if the collection is empty or if the given function returns false for at least one element it is applied to.

Overload 2

```
public:bool allElementsDo(IApplicator < Element >&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the collection until the applyTo() function returns false. The elements are visited in iteration order. Additional arguments may be passed as arguments to the constructor of the derived applicator class.

Notes:

1. The applyTo() function must not remove elements from or add elements to the collection. If you want to remove elements, you can use the removeAll() function with a property argument.
2. For the non-const version of allElementsDo(), the applyTo() function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to. Returns false if the collection is empty or if the applyTo() function returns false for at least one element it is applied to.

Overload 3

```
public:bool allElementsDo(IConstantApplicator < Element >&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the collection until the applyTo() function returns false. The elements are visited in iteration order. Additional arguments may be passed as arguments to the constructor of the derived applicator class.

Notes:

1. The applyTo() function must not remove elements from or add elements to the collection. If you want to remove elements, you can use the removeAll() function with a property argument.
2. For the non-const version of allElementsDo(), the applyTo() function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to. Returns false if the collection is empty or if the applyTo() function returns false for at least one element it is applied to.

Overload 4

```
public:bool
  allElementsDo( bool ( * applicatorFunction ) ( Element const&,
                 void * ) , void* additionalArgument = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements in the collection until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Notes:

1. The given function must not remove elements from or add elements to the collection. If you want to remove elements, you can use the removeAll() function with a property argument.
2. For the non-const version of allElementsDo(), the given function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the given function returns true for every element it is applied to. Returns false if the collection is empty or if the given function returns false for at least one element it is applied to.

Element Access

Member functions used to add, access or remove elements from the collection.

add

```
public:virtual bool add(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the collection is unique (with respect to elements of keys) and the element of key is already contained in the collection, sets the cursor to the existing element in the collection without adding the element. Otherwise, it adds the element to the collection and sets the cursor to the added element. In sequential collections, the given element is added as the last element. In sorted collections, the element is added at a position determined by the element or key value. Adding an element will either use the element's copy constructor or the assignment operator provided for the element type, depending on the implementation variant you choose.

Preconditions

- The cursor must belong to the collection.
- If the number is bounded and unique, the element of key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElement() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If an element as added, all cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Return Value

Returns true if the element was added.

Exceptions

- IOutOfMemory
- ICursorInvalidException
- IFullException
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addAllFrom

```
public:virtual void addAllFrom(IACollection < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds (copies) all elements of the given collection to the collection. The elements are added in the iteration order of the given collection. The contents of the elements, not the pointers to the elements, are copied. The elements are added according to the definition of add for this collection. The given collection is not changed.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual add operation:

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If an element as added, all cursors of this collection become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions

- IOutOfMemory
- IIdenticalCollectionException
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

any

```
public:Element const& any() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an arbitrary element of the collection.

Precondition

The collection must not be empty.

Exception

- IEmptyException

anyElement

```
public:Element const& anyElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an arbitrary element of the collection.

Precondition

The collection must not be empty.

removeAll

Removes all elements from this collection.

Overload 1

```
public:virtual INumber removeAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Element destructors are called as described in removeAt().

Side Effects

- All cursors of this collection become undefined.
- Collection classes supporting notifications send a modifyId notification.

Return Value

The number of elements removed.

Overload 2

```
public:virtual INumber
  removeAll( bool ( * predicateFunction ) ( Element const&,
             void * ) , void* additionalArgument = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this collection for which the given property function returns true. Additional arguments can be passed to the given property function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given. Element destructors are called as described in removeAt().

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

Return Value

The number of elements removed.

Miscellaneous

copy

```
public:virtual void copy(IACollection < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the given collection to this collection. Removes all elements from this collection, and adds the elements from the given collection. For more information on how adding is done, see addAllFrom().

Note: The given collection may be of a concrete type other than the collection itself. In this case, copying implicitly performs a conversion. If, for example, the given collection is a bag and the collection itself is a set, elements with multiple occurrences in the copied bag will only occur once in the resulting set.

Preconditions

Because the elements are copied one by one, the following preconditions are tested for each individual copy operation:

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- All cursors of this collection become undefined.
- If any elements were copied, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection has unique keys. This exception may be thrown, for example, when copying a bag into a map.

ImplOf

```
protected:static Implementation&
  ImplOf( IACollection < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notification Support

Member functions used to manage the notification capabilities of the collection.

disableNotification

```
public:INotifier& disableNotification()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the notifier from sending notifications to its observers.

enableNotification

```
public:INotifier& enableNotification(bool = true)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts the notifier sending notifications to its observers. This function can be overridden by derived classes to perform customized notification that your application might need. For instance, one of your function methods may require that a database be accessible before processing a retrieve function.

isEnabledForNotification

```
public:bool isEnabledForNotification() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is enabled for notification.

notifier

Returns the notifier object for this collection.

Overload 1

```
public:INotifier& notifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:INotifier const& notifier() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

notifyObservers

```
public:INotifier& notifyObservers(INotificationEvent const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers in a notifier's list of observers. Each observer receives a notification event containing the identity of the notifier, the notification ID, and any optional data provided by the specific notifier object.

Note:

A public and a protected version of notifyObservers are provided for convenience. The protected version does not require the caller to construct an INotificationEvent to call it. In this case, the construction of the INotificationEvent occurs in the code of the protected notifyObservers function.

operator INotifier &

```
public:operator INotifier &() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the notifier object for this collection.

Streaming Support

Member functions used to stream elements to and from data streams.

adoptStreamer

```
public:void adoptStreamer(IACollectionStreamer*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adopts a streamer object which is used to handle element streaming to and from a data stream.

operator <<=

```
public:virtual void operator <<=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in the stored elements to this collection. The adopted streamer object is used to do the streaming.

operator >>=

```
public:virtual void operator >>=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams the elements of this collection out to the passed data stream. The adopted streamer object is used to do the element streaming.

IACollection - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

Inherited Protected Data

None

IACollectionBase

Class header file: ibcllct.h

IACollectionBase - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IACollection

IACollectionBase - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct IACollectionBase objects.

~IACollectionBase

```
protected:~IACollectionBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IACollectionBase

Constructs an IACollectionBase object.

Overload 1

```
protected:IACollectionBase(IACollectionBase const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IACollectionBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Statics

CrsrImplOf

Overload 1

```
protected:static ICursorImpl const& CrsrImplOf(ICursor const& cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:static ICursorImpl& CrsrImplOf(ICursor& cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

InterfaceFor

```
protected:static ICursor* InterfaceFor(ICursorImpl* cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IACollectionBase - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IACollectionStreamer

Class header file: iastr.h

IACollectionStreamer - Hierarchy List

IACollectionStreamer

IACollectionStreamer - Member Functions and Data by Group

Constructors & Destructor

~IACollectionStreamer

```
public:virtual ~IACollectionStreamer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IACollectionStreamer

```
public:IACollectionStreamer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streaming

streamIn

```
public:virtual void
  streamIn( IDataStream& stream,
            void* element ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

streamOut

```
public:virtual void
  streamOut( IDataStream& stream,
             void* element ) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IACollectionStreamer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IADeque

A **deque**, or double-ended queue, is a sequence with restricted access. It is an ordered collection of elements with no key and no element equality. The elements are arranged so that each collection has a first and a last element, each element except the last has a next element, and each element but the first has a previous element. You can only add or remove the first or the last element.

The type and value of the elements are irrelevant, and have no effect on the behavior of the collection.

An example of using a deque is a program for managing a lettuce warehouse. Cases of lettuce arriving into the warehouse are registered at one end of the queue (the ″fresh″ end) by the receiving department. The shipping department reads the other end of the queue (the ″old″ end) to determine which case of lettuce to ship next. However, if an order comes in for very fresh lettuce, which is sold at a premium, the shipping department reads the ″fresh″ end of the queue to select the freshest case of lettuce available.

Class header file: iadqu.h

IADeque - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IADeque
        IGDequeAsList
        IGDequeAsDilTable
        IRDeque
        IGDeque
        IGDequeAsTable

IADeque - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IADeque

```
public:~IADeque()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IADeque

Constructs an IADeque object.

Overload 1

```
public:IADeque(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IADeque(IADeque < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IADeque()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Add Elements

Use these member functions to add elements to the collection.

addAsFirst

Adds the element to the collection as the first element in sequential order. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- ICursorInvalidException
- IOutOfMemory
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAsFirst(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsFirst(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

addAsLast

Adds the element to the collection as the last element in sequential order. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.

- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- ICursorInvalidException
- IOutOfMemory
- IFullException.

Overload 1

```
public:virtual void addAsLast(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsLast(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Comparison

Use these member functions as comparison operators.

compare

```
public:ioc::numeric_t
  compare( IADeque < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Remove Elements

Use these member functions to remove elements from the collection.

removeFirst

```
public:virtual void removeFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the first element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects

- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception

- IEmptyException

removeLast

```
public:virtual void removeLast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the last element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects

- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception

- IEmptyException

IADeque - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IAEqualityCollection

IAEqualityCollection is an abstract class. It cannot be used to create any objects. The IAEqualityCollection class defines the interfaces for the property of element equality.

The following abstract classes are derived from IAEqualityCollection:
- IAEqualityKeyCollection
- IAEqualitySortedCollection

- IAEqualitySortedCollection

The following concrete classes are defined by IAEqualityCollection:
- IBag and its variants
- IEqualitySequence and its variants
- ISet and its variants

Class header file: iaequal.h

IAEqualityCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAEqualitySequence
IABag
IAEqualityKeySortedCollection
IAEqualitySortedCollection
IASet

IAEqualityCollection - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

~IAEqualityCollection

```
public:~IAEqualityCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAEqualityCollection

Constructs an IAEqualityCollection object.

Overload 1

```
public:IAEqualityCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a

collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAEqualityCollection( IAEqualityCollection < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAEqualityCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Miscellaneous

These member functions perform miscellaneous operations.

contains

```
public:bool contains(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection contains an element equal to the given element.

containsAllFrom

```
public:bool containsAllFrom(IACollection < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all the elements of the given collection are contained in the collection. The definition of containment is described in IAEqualityCollection::contains()

locate

```
public:bool locate(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates an element in the collection that is equal to the given element. Sets the cursor to point to the element in the collection, or invalidates the cursor if no such element exists.

If the collection contains several such elements, the first element in iteration order is located.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if an element was found.

Exceptions

- ICursorInvalidException

locateNext

```
public:bool locateNext(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection that is equal to the given element, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of the given element are left to be visited.

Note: If you code a call to IACollection::locateFirst() and a set of calls to IASet::locateNext(), each occurrence of an element will be visited exactly once in iteration order.

Precondition

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locateOrAdd

Locates an element in the collection that is equal to the given element. (See IAEqualityCollection::locate() for details on locate().) If no such element is found, locateOrAdd() adds the element as described in IACollection::add(). The cursor is set to the located or added element.

Note: This method may be more efficient than using locate() followed by a conditionally called IACollection::add().

Preconditions
- The cursor must belong to the collection.
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.
- The element or key must exist, or (IACollection::maxNumberOfElements()).

Side Effects
- If the element was added, all cursors of this collection, except the given cursor, become undefined.
- If the element was added, collection classes supporting notifications send an addId notification.

Return Value

Returns true if the element was located. Returns false if the element could not be located but had to be added.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map

Overload 1

```
public:virtual bool locateOrAdd(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool locateOrAdd(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

numberOfOccurrences

```
public:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

remove

```
public:virtual bool remove(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes an element in the collection that is equal to the given element. If no such element exists, the collection remains unchanged. In collections with nonunique elements, an arbitrary occurrence of the given element will be removed. Element destructors are called as described in IACollection::removeAt().

Side Effects
- If an element was removed, all cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Return Value

Returns true if an element was removed.

removeAllOccurrences

```
public:virtual INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in IACollection::removeAt().

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

If any elements were removed, all cursors of this collection become undefined.

IAEqualityCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

Inherited Protected Data

None

IAEqualityKeyCollection

IAEqualityKeyCollection is an abstract class. It cannot be used to create any objects. It defines the interfaces for the following properties:
- Element equality
- Key equality

The following concrete classes are defined by IAEqualityKeyCollection:
- IMap and its variants
- IRelation and its variants

Class header file: iaeqkey.h

IAEqualityKeyCollection - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection
        IAEqualityKeyCollection
          IAMap
          IARelation

IAEqualityKeyCollection - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

~IAEqualityKeyCollection

```
public:~IAEqualityKeyCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAEqualityKeyCollection

Constructs an IAEqualityKeyCollection object.

Overload 1

```
public:IAEqualityKeyCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAEqualityKeyCollection( IAEqualityKeyCollection < Element,
                          Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAEqualityKeyCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the

required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IAEqualityKeyCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IAEqualityKeySortedCollection

IAEqualityKeySortedCollection is an abstract class that defines the interfaces for the following properties:
- Element equality
- Key equality
- Sorted elements

The following concrete classes are defined by IAEqualityKeySortedCollection:
- ISortedMap and its variants
- ISortedRelation and its variants

Class header file: iaeqksrt.h

IAEqualityKeySortedCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IASortedMap

IAEqualityKeySortedCollection - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

~IAEqualityKeySortedCollection

```
public:~IAEqualityKeySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAEqualityKeySortedCollection

Constructs an IAEqualityKeySortedCollection object.

Overload 1

```
public:IAEqualityKeySortedCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAEqualityKeySortedCollection( IAEqualityKeySortedCollection < Element,
                                         Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAEqualityKeySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IAEqualityKeySortedCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IAEqualitySequence

An **equality sequence** is an ordered collection of elements. The elements are arranged so that each collection has a first and a last element, each element except the last has a next element, and each element but the first has a previous element. An equality sequence supports element equality, which gives you the ability, for example, to search for particular elements.

An example of using an equality sequence is a program that calculates members of the Fibonacci sequence and places them in a collection. Multiple elements of the same value are allowed. For example, the sequence begins with two instances of the value 1. You can search for a given element, for example 8, and find out what element follows it in the sequence. Element equality allows you to do this, using the locate() and setToNext() functions.

Class header file: iaes.h

IAEqualitySequence - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection
        IAEqualitySequence
          IGEqualitySequenceAsTable
          IREqualitySequence

IGEqualitySequenceAsList
IGEqualitySequence
IGEqualitySequenceAsDilTable

IAEqualitySequence - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of the class.

~IAEqualitySequence

```
public:~IAEqualitySequence()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAEqualitySequence

Constructs an IAEqualitySequence object.

Overload 1

```
public:IAEqualitySequence(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAEqualitySequence(IAEqualitySequence < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAEqualitySequence()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions as comparison operators.

compare

```
public:ioc::numeric_t
  compare( IAEqualitySequence < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

locateFirst

```
public:bool locateFirst(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the first element in iteration order in the collection that is equal to the given element. Sets the cursor to the located element, or invalidates the cursor if no such element exists.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locateLast

```
public:bool locateLast(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the last element in iteration order in the collection that is equal to the given element. Sets the cursor to the located element, or invalidates the cursor if no such element exists.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locatePrevious

```
public:bool locatePrevious(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the previous element in iteration order that is equal to the given element, beginning at the element previous to the one specified by the given cursor and moving in reverse iteration order through the elements. Sets the cursor to the located element, or invalidates the cursor if no such element exists.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exceptions

- ICursorInvalidException

operator !=

```
public:bool
  operator !=( IAEqualitySequence < Element > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor does not point to the same element (of the same collection) as the given cursor.

operator ==

```
public:bool
  operator ==( IAEqualitySequence < Element > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to the same element (of the same collection) as the given cursor.

IAEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IAEqualitySortedCollection

IAEqualitySortedCollection is an abstract class. It cannot be used to create any objects. It defines the interfaces for the following properties:

- Element equality
- Sorted elements

The following concrete classes are defined by IAEqualitySortedCollection:

- ISortedBag and its variants
- ISortedSet and its variants

Class header file: iaeqsrt.h

IAEqualitySortedCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IASortedSet

IAEqualitySortedCollection - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAEqualitySortedCollection

```
public:~IAEqualitySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAEqualitySortedCollection

Constructs an IAEqualitySortedCollection object.

Overload 1

```
public:IAEqualitySortedCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAEqualitySortedCollection( IAEqualitySortedCollection
                                      < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAEqualitySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IAEqualitySortedCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IAHeap

A **heap** is an unordered collection of zero or more elements with no key. Element equality is not supported. Multiple elements are supported. The type and value of the elements are irrelevant, and have no effect on the behaviour of the heap.

You can compare using a heap collection to managing the scrap metal entering a scrapyard. Pieces of scrap are placed in the heapin an arbitrary location, and an element can be added multiple times (for example, the rear left fender from a particular kind of car). When a customer requests a certain amount of scrap, elements are removed from the heap in an arbitrary order until the required amount is reached. You cannot search for a specific piece of scrap except by examining each piece of scrap in the heap and manually comparing it to the piece you are looking for.

Class header file: iahp.h

IAHeap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAHeap

IAHeap - Member Functions and Data by Group

Constructors & Destructor

These members construct and destruct an IAHeap object.

~IAHeap

```
public:~IAHeap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAHeap

Constructs an IAHeap object.

Overload 1

```
public:IAHeap(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAHeap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 3

```
protected:IAHeap(IAHeap < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

IAHeap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

Inherited Protected Data

None

IAKeyBag

A **key bag** is an unordered collection of zero or more elements that have a key. Multiple elements are supported.

An example of using a key bag is a program that manages the distribution of combination locks to mambers of a fitness club. The element key is the number that is printed on the back of each combination lock. Each element also has data members for the club member's name, member number, and so on. When you join the club, you are given one of the available combination locks, and your name, menber number, and the number on the combination lock are entered into the collection. Because a given number on a combination locak may appear on several locks, the program allows the same lock number to be added to the collection multiple times. When you return a lock, the program finds each element whose key matches your lock's serial number, and deletes one such element that has your name associated with it.

Class header file: iakb.h

IAKeyBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsTable
IGKeyBagAsList
IGKeyBagAsHshTable
IRKeyBag
IGKeyBagAsDilTable
IGKeyBag

IAKeyBag - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IAKeyBag objects.

~IAKeyBag

```
public:~IAKeyBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeyBag

Constructs an IAKeyBag object.

Overload 1

```
public:IAKeyBag(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAKeyBag(IAKeyBag < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeyBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IAKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

Inherited Protected Data

None

IAKeyCollection

IAKeyCollection is an abstract class. It cannot be used to create any objects. The IAKeyCollection class inherits from IACollection and defines the interfaces for the key property.

The following abstract classe are derived from IAKeyCollection:
- IAEqualityKeyCollection
- IAEqualityKeySortedCollection
- IAKeySortedCollection

The following concrete classes are defined by IAKeyCollection:
- IKeyBag and its variants
- IKeySet and its variants

Class header file: iakey.h

IAKeyCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IAKeyBag
IAKeySortedCollection

IAKeyCollection - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAKeyCollection

```
public:~IAKeyCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeyCollection

Constructs and object of this class.

Overload 1

```
public:IAKeyCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAKeyCollection(IAKeyCollection < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeyCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Miscellaneous

Use these member functions to perform miscellaneous operations.

addOrReplaceElementWithKey

If an element is contained in the collection where the key is equal to the key of the given element, sets the cursor to this element in the collection and replaces it with the given element. Otherwise, it adds the given element to the collection, and sets the cursor to the added element. If the given element is added, the contents of the element, not a pointer to it, is added.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, an element with the given key must be contained in the collection, or (numberOfElements() < maxNumberOfElements()).

Side Effects
- If the element was added, all cursors of this collection, except the given cursor, become undefined.
- If the element was added, collection classes supporting notifications send a replaceId notification.

Return Value

Returns true if the element was added. Returns false if the element was replaced.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual bool
  addOrReplaceElementWithKey( Element const&,
                              ICursor& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool addOrReplaceElementWithKey(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

containsAllKeysFrom

```
public:bool
  containsAllKeysFrom( IACollection < Element > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all of the keys of the given collection are contained in the collection.

containsElementWithKey

```
public:bool containsElementWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection contains an element with the same key as the given key.

elementWithKey

Returns a reference to an element specified by the key.

Note:

1. For the version of IAKeyCollection::elementWithKey **without** a **const** suffix, do not manipulate the element in the collection in a way that changes the positioning property of the element.
2. If there are several elements with the given key, an arbitrary one is returned.

Precondition

The given key is contained in the collection.

Exception

- INotContainsKeyException

Overload 1

```
public:Element& elementWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element const& elementWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

key

```
public:Key const& key(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the key of the given element using the IAKeyCollection::key function provided for the element type.

locateElementWithKey

```
public:bool locateElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates an element in the collection with the same key as the given key. Sets the cursor to point to the element in the collection, or invalidates the cursor if no such element exists.

If the collection contains several such elements, the first element in iteration order is located.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locateNextElementWithKey

```
public:bool locateNextElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection with the given key, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of such an element are left to be visited.

Note: If you code a call to IACollection::locateFirst and a set of calls to locateNextElementWithKey(), each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locateOrAddElementWithKey

Locates an element in the collection with the given key as described for the IACollection::locateElementWithKey function. If no such element exists, IACollection::locateOrAddElementWithKey adds the element as described in IACollection::add. The cursor is set to the located or added element.

Preconditions
- If the collection is bounded and an element with the given key is not already contained, (numberOfElements() < maxNumberOfElements()).
- The cursor must belong to the collection.

Side Effects
- If the element was added, all cursors of this collection, except the given cursor, become undefined.
- If the element was added, collection classes supporting notifications send a addId notification.

Return Value

Returns true if the element was located. Returns false if the element could not be located but had to be added.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual bool locateOrAddElementWithKey(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool
  locateOrAddElementWithKey( Element const&,
                             ICursor& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

numberOfDifferentKeys

```
public:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
public:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

removeAllElementsWithKey

```
public:virtual INumber removeAllElementsWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection with the same key as the given key. Element destructors are called as described in IACollection::removeAt.

Side Effects
- If any elements were removed, all cursors of this collection become undefined.

- If any elements were removed, collection classes supporting notifications send a removeId notification.

Return Value

The number of elements removed.

removeElementWithKey

```
public:virtual bool removeElementWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes an element from the collection with the same key as the given key. If no such element exists, the collection remains unchanged. In collections with nonunique elements, an arbitrary occurrence of such an element will be removed. Element destructors are called as described in IACollection::removeAt.

Side Effects

- If an element was removed, all cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Return Value

Returns true if an element was removed.

replaceElementWithKey

Replaces an element with the same key as the given element by the given element, and sets the cursor to this element. If no such element exists, it invalidates the cursor. In collections with nonunique elements, an arbitrary occurrence of such an element will be replaced.

Precondition

The cursor must belong to the collection.

Side Effects

- Collection classes supporting notifications send a replaceId notification.

Return Value

Returns true if an element was replaced.

Exceptions

- ICursorInvalidException

Overload 1

```
public:virtual bool
  replaceElementWithKey( Element const&,
                         ICursor& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool replaceElementWithKey(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

setToNextWithDifferentKey

```
public:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception
- ICursorInvalidException

IAKeyCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

Inherited Protected Data

None

IAKeySet

A **key set** is an unordered collection of zero or more elements that have a key. Element equality is not supported. Only unique elements are supported, in terms of their key.

An example of using a key set is a program that allocates rooms to patrons checking into a hotel. The room number serves as the element's key, and the patron's name is a data member of the element. When you check in at the front desk, the clerk pulls a room key from the board, and enters that key's number and your name into the collection. When you return the key at check-out time, the record for that key is removed from the collection. You cannot add an element to the collection that is already present, becuase there is only one key for each room. If you attempt to add an element that is already present, the add() function returns false to indicate that the element was not added.

Class header file: iaks.h

IAKeySet - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - **IAKeySet**
        - IGKeySetAsAvlTree
        - IGKeySetAsBstTree
        - IGKeySetAsHshTable
        - IGKeySetAsTable
        - IRKeySet
        - IGKeySet
        - IGKeySetAsList
        - IGKeySetAsDilTable

IAKeySet - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IAKeySet objects.

~IAKeySet

```
public:~IAKeySet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeySet

Constructs an IAKeySet object.

Overload 1

```
public:IAKeySet(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAKeySet(IAKeySet < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeySet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Key Operations

Locate elements with a particular key, find out how many different keys are available, or remove all elements with a particular key.

numberOfDifferentKeys

```
protected:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
protected:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

removeAllElementsWithKey

```
protected:INumber removeAllElementsWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection with the same key as the given key. Element destructors are called as described in IACollection::removeAt.

Side Effects
- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a removeId notification.

Return Value

The number of elements removed.

Location and Iteration

Locate elements that match or do not match a particular key.

locateNextElementWithKey

```
protected:bool locateNextElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection with the given key, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of such an element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to IAPriorityQueue::locateNextElementWithKey, each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception
- ICursorInvalidException

setToNextWithDifferentKey

```
protected:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception
- ICursorInvalidException

IAKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAKeyCollection`	909		

Inherited Protected Data

None

IAKeySortedBag

A **key sorted bag** is an ordered collection of zero or more elements that have a key. Elements are sorted according to the value of their key field. Element equality is not supported. Multiple elements are supported.

An example of using a key sorted bag is a program that maintains a list of families, sorted by the number of family members in each family. The key is the number of family members. You can add an element whose key is already in the collection (because two families can have the same number of members), and you can generate a list of families sorted by size. You cannot locate a family except by its key, because a key sorted bag does not support element equality.

Class header file: iaksb.h

IAKeySortedBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeySortedCollection
        - **IAKeySortedBag**
          - IGKeySortedBag
          - IGKeySortedBagAsTable
          - IRKeySortedBag
          - IGKeySortedBagAsList
          - IGKeySortedBagAsDilTable

IAKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAKeySortedBag

```
public:~IAKeySortedBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeySortedBag

Constructs an IAKeySortedBag object.

Overload 1

```
public:IAKeySortedBag(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAKeySortedBag(IAKeySortedBag < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeySortedBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions to compare two collections.

compare

```
public:ioc::numeric_t
  compare( IAKeySortedBag < Element,
           Key > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*,IACollection::compare is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own IACollection::compare function as described in Using Separate Functions in the **Class Library User's Guide**.

Return Value

Returns the result of the collection comparison.

IAKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

Inherited Protected Data

None

IAKeySortedCollection

IAKeySortedCollection is an abstract class. It cannot be used to create any objects. The IAKeySortedCollection class inherits from the IASortedCollection and IAKeyCollection classes. It defines the interfaces for the following properties:
- Key equality
- Sorted elements

The following concrete classes are defined by IAKeySortedCollection:
- IKeySortedBag and its variants
- IKeySortedSet and its variants

Class header file: iaksrt.h

IAKeySortedCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IAKeySortedSet

IAKeySortedCollection - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.
~IAKeySortedCollection

```
public:~IAKeySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeySortedCollection

Constructs an IAKeySortedCollection object.

Overload 1

```
public:IAKeySortedCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAKeySortedCollection( IAKeySortedCollection < Element,
                        Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeySortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked

when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IAKeySortedCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

Inherited Protected Data

None

IAKeySortedSet

A **key sorted set** is an ordered collection of zero or more elements that have a key. Elements are sorted according to the value of their key field. Element equality is not supported. Only elements with unique keys are supported. A request to add an element whose key already exists is ignored.

An example of using a key sorted set is a program that keeps track of canceled credit card numbers and the individuals to whom they are issued. Each card number occurs only once, and the collection is sorted by card number. When a merchant enters a customer's card number into a point-of-sale terminal, the collection is checked to see if that card number is listed in the collection of canceled cards. If it is found, the name of the individual is shown, and the merchant is given directions for contacting the card company. If the card number is not found, the transaction can proceed because the card is valid. A list of canceled cards is printed out each month, sorted by card number, and distributed to all merchants who do not have an automatic point-of-sale terminal installed.

Class header file: iakss.h

IAKeySortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet

IGKeySortedSetAsBstTree
IRKeySortedSet
IGKeySortedSetAsTable
IGKeySortedSetAsAvlTree
IGKeySortedSetAsList
IGKeySortedSetAsDilTable
IGKeySortedSet

IAKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAKeySortedSet

```
public:~IAKeySortedSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAKeySortedSet

Constructs an IAKeySortedSet object.

Overload 1

```
public:IAKeySortedSet(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAKeySortedSet(IAKeySortedSet < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAKeySortedSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparisons

Use this member function to compare two collections.

compare

```
public:ioc::numeric_t
  compare( IAKeySortedSet < Element,
           Key > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous functions.

numberOfDifferentKeys

```
protected:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
protected:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

removeAllElementsWithKey

```
protected:INumber removeAllElementsWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection with the same key as the given key. Element destructors are called as described in IACollection::removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a removeId notification.

Return Value

The number of elements removed.

setToNextWithDifferentKey

```
protected:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception

- ICursorInvalidException

IAKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

Inherited Protected Data

None

IAMap

A **map** is an unordered collection of zero or more elements that have a key. Element equality is supported and the values of the elements are relevant.

Only elements with unique keys are supported. A request to add an element whose key already exists in another element of the collection causes an exception to be thrown. A request to add a duplicate element is ignored.

An example of using a map is a program that translates integer values between the ranges of 0 and 20 to their written equivalents, or between written numbers and their numeric values. Two maps are created, one with the integer values as keys, one with the written equivalents as keys. You can enter a number, and that number is used as a key to locate the written equivalent. You can enter a written equivalent of a number, and that text is used as a key to locate the value. A given key always matches only one element. You cannot add an element with a key of 1 or ″one″ if that element is already present in the collection.

Class header file: iamap.h

IAMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsList
IGMapAsDilTable
IRMap
IGMap
IGMapAsBstTree
IGMapAsTable
IGMapAsHshTable
IGMapAsAvlTree

IAMap - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAMap

```
public:~IAMap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAMap

Constructs an IAMap object.

Overload 1

```
public:IAMap(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAMap(IAMap < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAMap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Miscellaneous

Use these members to perform miscellaneous operations.

addDifference

```
public:virtual void
  addDifference( IAMap < Element,
                 Key > const&,
                 IAMap < Element,
                 Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IABag::differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IAMap < Element,
                   Key > const&,
                   IAMap < Element,
                   Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IABag::intersectionWith..

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IAMap < Element,
            Key > const&,
            IAMap < Element,
            Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IABag::unionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

differenceWith

```
public:virtual void differenceWith(IAMap < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times,the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IAMap < Element,
                    Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

operator !=

```
public:bool operator !=(IAMap < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see ICursor::operator ==.

operator ==

```
public:bool operator ==(IAMap < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Condition
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

unionWith

```
public:virtual void unionWith(IAMap < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

locateNext

```
protected:bool locateNext(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection that is equal to the given element, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of the given element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to IAMap::locateNext, each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

locateNextElementWithKey

```
protected:bool locateNextElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection with the given key, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of such an element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to IAKeySet::locateNextElementWithKey, each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

numberOfDifferentKeys

```
protected:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
protected:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

removeAllElementsWithKey

```
protected:INumber removeAllElementsWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection with the same key as the given key. Element destructors are called as described in IACollection::removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a removeId notification.

Return Value

The number of elements removed.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in IACollection::removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

setToNextWithDifferentKey

```
protected:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception

- ICursorInvalidException

IAMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityKeyCollection`	889		

Inherited Protected Data

None

IAMultiwayTree

A multiway tree, also known as an **n-ary tree**, is a special tree where each node can have up to **n** children.

n must be greater than one. If **n** is one, the tree is a list. If **n** is zero, the structure loses its meaning.

An example of using an multiway tree is a program used to build a family tree. (For simplicity, assume that the family tree is not concerned with information about spouses.) Whenever you discover a relative who is not already in your family tree, you enter the relative's name. If you know the parent's name, and the parent is already in the collection, the new relative is added as a child of the existing parent. If the parent is known but is not in the collection, a new collection is created, with the parent as the root element and the child as a child node of the parent. If you do not know the parent, the relative is entered as the root element of a new collection. You can also enter information about the children of a given relative; this information is used to attach a subtree, whose root node is the child, to the node of the parent of that child. Once you have established the collection, you can determine who is the parent or oldest known ancestor of a given relative, and you can display a list of all descendents of a given family member.

Class header file: iamwt.h

IAMultiwayTree - Hierarchy List

IMetaTree
  IATreeBase
    IATree
      IAMultiwayTree
        IGMultiwayTree

IAMultiwayTree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct objects of the class.

IAMultiwayTree

Constructs an IAMultiway Tree.

Overload 1

```
protected:IAMultiwayTree( IAMultiwayTree < numOfChildren,
                Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree by copying all elements from the given tree.

Exception
- IOutOfMemory

Overload 2

```
protected:IAMultiwayTree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree. The tree is initially empty; that is, it does not contain any nodes.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

attachAsChild

```
public:void
  attachAsChild( ITreeCursor const&,
                 IPosition,
                 IAMultiwayTree < numOfChildren,
                 Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the subtree denoted by the given subtree cursor as a child with the given position of the node (of this tree) denoted by the given cursor. Removes this subtree from the given tree. The attachAsChild() function (without a subtree cursor argument) copies and removes the whole given tree.

Be careful when this tree and the given tree are the same. In such cases you must not attach a subtree to one of its own children, because a cyclic tree structure would result. Because attachSubtreeAsChild() removes this subtree from this tree, you will never be able to access either this subtree or the given subtree attached to it. This practice can also lead to memory not being properly freed.

This warning applies to both attachAsChild() and attachSubtreeAsChild().

Note: These functions are implemented by copying a pointer to the subtree, rather than by copying all elements in the subtree.

Preconditions
- The cursor must point to an element of this tree.
- The subtree cursor must point to an element of the given tree.
- (1 <= **position** <= **numberOfChildren**).
- The node denoted by the given cursor (of this tree) must not have a child at the given position.
- If this tree and the given tree are the same, a subtree must not be attached to one of its own children.

Exceptions
- ICursorInvalidException
- IPositionInvalidException
- IChildAlreadyExistsException
- ICyclicAttachException

attachAsRoot

```
public:void
  attachAsRoot( IAMultiwayTree < numOfChildren,
                Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the subtree denoted by the cursor of the given tree to (the root of) this tree, and removes this subtree from the given tree. The attachAsRoot() function (without a cursor argument) copies and removes the whole given tree.

Note: These functions are implemented by copying a pointer to the subtree, rather than by copying all elements in the subtree.

Preconditions
- The cursor must point to an element of this tree.
- The tree must not have a root; that is, it must be empty.

Exceptions
- ICursorInvalidException
- IRootAlreadyExistsException

attachSubtreeAsChild

```
public:void
  attachSubtreeAsChild( ITreeCursor const&,
                        IPosition,
                        IAMultiwayTree < numOfChildren,
                        Element > const&,
                        ITreeCursor const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the subtree denoted by the given subtree cursor as a child with the given position of the node (of this tree) denoted by the given cursor. Removes this subtree from the given tree. The attachAsChild() function (without a subtree cursor argument) copies and removes the whole given tree.

Be careful when this tree and the given tree are the same. In such cases you must not attach a subtree to one of its own children, because a cyclic tree structure would result. Because attachSubtreeAsChild() removes this subtree from this tree, you will never be able to access either this subtree or the given subtree attached to it. This practice can also lead to memory not being properly freed.

This warning applies to both attachAsChild() and attachSubtreeAsChild().

Note: These functions are implemented by copying a pointer to the subtree, rather than by copying all elements in the subtree.

Preconditions
- The cursor must point to an element of this tree.

- The subtree cursor must point to an element of the given tree.
- (1 <= **position** <= **numberOfChildren**:).
- The node denoted by the given cursor (of this tree) must not have a child at the given position.
- If this tree and the given tree are the same, a subtree must not be attached to one of its own children.

Exceptions
- ICursorInvalidException
- IPositionInvalidException
- IChildAlreadyExistsException
- ICyclicAttachException

attachSubtreeAsRoot

```
public:void
  attachSubtreeAsRoot( IAMultiwayTree < numOfChildren,
                       Element > const&,
                       ITreeCursor const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the subtree denoted by the cursor of the given tree to (the root of) this tree, and removes this subtree from the given tree. The attachAsRoot() function (without a cursor argument) copies and removes the whole given tree.

Note: These functions are implemented by copying a pointer to the subtree, rather than by copying all elements in the subtree.

Preconditions
- The cursor must point to an element of this tree.
- The tree must not have a root; that is, it must be empty.

Exceptions
- ICursorInvalidException
- IRootAlreadyExistsException

copy

```
public:void copy(IAMultiwayTree < numOfChildren, Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this tree, and copies the subtree denoted by the given cursor of the given tree to (the root of) this tree. The copy function (without a cursor argument) copies the whole given tree.

Preconditions

The cursor must point to an element of the given tree.

Exceptions
- IOutOfMemory
- ICursorInvalidException

copySubtree

```
public:void
  copySubtree( IAMultiwayTree < numOfChildren,
               Element > const&,
               ITreeCursor const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this tree, and copies the subtree denoted by the given cursor of the given tree to (the root of) this tree. The copy function (without a cursor argument) copies the whole given tree.

Preconditions

The cursor must point to an element of the given tree.

Exceptions
- IOutOfMemory
- ICursorInvalidException

hasChild

```
public:bool hasChild(IPosition, ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the node pointed to by the given cursor has a child at the given position.

Preconditions
- The cursor must point to an element of this tree.
- (1 <= **position** <= **numberOfChildren**)

Exceptions
- ICursorInvalidException
- IPositionInvalidException

numberOfChildren

```
public:INumber numberOfChildren() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of children a node can possibly have. The actual number of children of any node will always be less than or equal to this number.

IAMultiwayTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

Inherited Protected Data

None

IAOrderedCollection

IAOrderedCollection is an abstract class. It cannot be used to create any objects. IAOrderedCollection defines the interfaces for the property of ordered elements.

The following abstract classes are derived from IAOrderedCollection:
- IASortedCollection
- IASequentialCollection

Class header file: iaorder.h

IAOrderedCollection - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAOrderedCollection
        IASequentialCollection

IAOrderedCollection - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAOrderedCollection

```
public:~IAOrderedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAOrderedCollection

Constructs an IAOrderedCollection object.

Overload 1

```
public:IAOrderedCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAOrderedCollection(IAOrderedCollection < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAOrderedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Cursor Support

Set cursors to point at particular elements of the collection.

setToLast

```
public:bool setToLast(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last element of the collection in iteration order. If the collection is empty (if no last element exists), the given cursor is no longer valid.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if the collection is not empty.

Exceptions
- ICursorInvalidException

setToPosition

```
public:void setToPosition(IPosition, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the element at the given position. Position 1 specifies the first element.

Precondition
- The cursor must belong to the collection.
- Position must be a valid position in the collection; that is, (1 <= **position** <= numberOfElements()).

Exceptions
- ICursorInvalidException
- IPositionInvalidException

setToPrevious

```
public:bool setToPrevious(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous element in iteration order, or invalidates the cursor if no such element exists.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a previous element exists.

Exception
- ICursorInvalidException

Element Access

Access particular elments in the collection.

elementAtPosition

```
public:Element const& elementAtPosition(IPosition) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the element at the given position in the collection.

Position 1 specifies the first element.

Position must be a valid position in the collection; that is, (1 <= **position** <= numberOfElements()).

Precondition

(1 <= **position** <= numberOfElements()).

Exception
- IPositionInvalidException

first

```
public:Element const& first() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as firstElement().

firstElement

```
public:Element const& firstElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the first element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

last

```
public:Element const& last() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as lastElement().

lastElement

```
public:Element const& lastElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the last element of the collection.

Precondition

The collection must not be empty.

Exception

- IEmptyException

Element Position Testing

Tests if a cursor is pointing at an element at a particular position in the collection.

isFirst

```
public:bool isFirst(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as isFirstAt().

isFirstAt

```
public:bool isFirstAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given cursor points to the first element of the collection.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Exception

- ICursorInvalidException

isLast

```
public:bool isLast(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as isLastAt().

isLastAt

```
public:bool isLastAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given cursor points to the last element of the collection.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Exception
- ICursorInvalidException

position

```
public:IPosition position(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as positionAt().

positionAt

```
public:IPosition positionAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines the position of the current element. position 1 specifies the position of the first element.

Preconditions

The cursor must belong to the collection, and the cursor must point to an element of the collection.

Exceptions
- ICursorInvalidException

Remove Elements

Remove elements from the collection.

removeAtPosition

```
public:virtual void removeAtPosition(IPosition)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the element from the collection that is at the given position. Element destructors are called as described in IACollection::removeAt.

The first element of the collection has position 1.

Precondition

Position must be a valid position in the collection; that is, (1 <= **position** <= numberOfElements()).

Side Effects
- All cursors of this collection become undefined.
- Collection classes supporting notifications send a removeId notification.

Exception
- IPositionInvalidException

removeFirst

```
public:virtual void removeFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the first element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception
- IEmptyException

removeLast

```
public:virtual void removeLast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the last element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception
- IEmptyException

IAOrderedCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

Inherited Protected Data

None

IApplicator

The classes IApplicator and IConstantApplicator define the interface for applicator objects. The redefinition of the function applyTo() defines the actions that are performed with the version of allElementsDo() that takes an applicator argument. Iteration stops when applyTo() returns false.

Class header file: iappl.h

IApplicator - Hierarchy List

IApplicator

IApplicator - Member Functions and Data by Group

Apply To

The applyTo function.

applyTo

```
public:virtual bool applyTo(Element&) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This functions applies a series of specified statements or a function to all elements of a collection for which you use the applicator. For example, myCollection.allElementsDo(myApplicator); causes the code in the applyTo() function that you code for your applicator object myApplicator to be applied to all elements of the collection myCollection.

IApplicator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IAPriorityQueue

A **priority queue** is a key sorted bag with restricted access. It is an ordered collection of zero or more elements. Keys and multiple elements are supported. Element equality is not supported.

When an element is added, it is placed in the queue according to its key value or priority. The highest priority is indicated by the lowest key value. You can remove the element with the highest priority. Within the priority queue, elements are sorted according to ascending key values, as in other key collections. You can only remove the element with the lowest key value.

For elements with equal priority, the priority queue has a first-in, first-out behavior.

An example of a priority queue is a program used to assign priorities to service calls in a heating repair firm. When a customer calls with a problem, a record with the customer's name and the seriousness of the situation is placed in a priority queue. When a service person becomes available, customers are chosen by the program beginning with those whose situation is most severe. In this example, a serious problem such as a nonfunctioning furnace would be indicated by a low value for the priority, and a minor problem such as a noisy radiator would be indicated by a high value for the priority.

Class header file: iapqu.h

IAPriorityQueue - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAPriorityQueue
        IGPriorityQueue
        IRPriorityQueue
        IGPriorityQueueAsTable
        IGPriorityQueueAsList

IAPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAPriorityQueue

```
public:~IAPriorityQueue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAPriorityQueue

Constructs an IAPriorityQueue object.

Overload 1

```
public:IAPriorityQueue(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAPriorityQueue(IAPriorityQueue < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAPriorityQueue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions to compare a collection with a given collection.

compare

```
public:ioc::numeric_t
  compare( IAPriorityQueue < Element,
           Key > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Element Addition/Removal

Use these members to add, remove or reorder elements within a collection.

dequeue

Copies the first element of the collection to the given element, and removes it from the collection.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If the element is removed, collection classes supporting notifications send a modifyId notification.

Exception
- IEmptyException

Overload 1

```
public:virtual void dequeue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void dequeue(Element&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

enqueue

Adds the element to the collection, and sets the cursor to the added element. For ordinary queues, the given element is added as the last element. For priority queues, the element is added at a position determined by the ordering relation provided for the element or key type.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection except the given cursor become undefined.
- If the element is added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void enqueue(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void enqueue(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

locateElementWithKey

```
public:bool locateElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates an element in the collection with the same key as the given key. Sets the cursor to point to the element in the collection, or invalidates the cursor if no such element exists.

If the collection contains several such elements, the first element in iteration order is located.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if an element was found.

Exception
- ICursorInvalidException

locateNextElementWithKey

```
public:bool locateNextElementWithKey(Key const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection with the given key, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of such an element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to locateNextElementWithKey(), each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception
- ICursorInvalidException

locateOrAddElementWithKey

Locates an element in the collection with the given key as described for the IACollection::locateElementWithKey function. If no such element exists, locateOrAddElementWithKey() adds the element as described in IACollection::add. The cursor is set to the located or added element.

Preconditions
- If the collection is bounded and an element with the given key is not already contained, (numberOfElements() < maxNumberOfElements()).
- The cursor must belong to the collection.

Side Effects

- If the element was added, all cursors of this collection, except the given cursor, become undefined.
- If the element is added, collection classes supporting notifications send a addId notification.

Return Value

Returns true if the element was located. Returns false if the element could not be located but had to be added.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual bool
  locateOrAddElementWithKey( Element const&,
                             ICursor& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool locateOrAddElementWithKey(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

removeFirst

```
public:virtual void removeFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the first element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception
- IEmptyException

Miscellaneous

Use these member functions to perform miscellaneous functions.

containsAllKeysFrom

```
public:bool
  containsAllKeysFrom( IARestrictedAccessCollection < Element > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if all of the keys of the given collection are contained in the collection.

containsElementWithKey

```
public:bool containsElementWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection contains an element with the same key as the given key.

elementWithKey

```
public:Element const& elementWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an element specified by the key.

Note:

1. For the version of elementWithKey() **without** a **const** suffix, do not manipulate the element in the collection in a way that changes the positioning property of the element.
2. If there are several elements with the given key, an arbitrary one is returned.

Precondition

The given key is contained in the collection.

Exception

- INotContainsKeyException

key

```
public:Key const& key(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the key of the given element using the key() function provided for the element type.

numberOfDifferentKeys

```
public:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
public:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

setToNextWithDifferentKey

```
public:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception
- ICursorInvalidException

IAPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IAQueue

A **queue** is a sequence with restricted access. It is an ordered collection of elements with no key and no element equality. The elements are arranged so that each collection has a first and a last element, each element except the last has a next element, and each element but the first has a previous element. The type and value of the elements are irrelevant, and have no effect on the behavior of the collection.

You can only add an element as the last element, and you can only remove the first element. Consequently, The elements of a queue are in chronological order.

A queue is characterized by a first-in, first-out (FIFO) behaviour.

An example of using a queue is a program that processes requests for parts at the cash sales desk of a warehouse. A request for a part is added to the queue when the customer's order is taken, and is removed from the queue when an order picker receives the order form for the part. Using a queue collection in such an application ensures that all orders for parts are processed on a first-come, first-served basis.

Class header file: iaqu.h

IAQueue - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAQueue
        IGQueue
        IGQueueAsTable
        IRQueue
        IGQueueAsList
        IGQueueAsDilTable

IAQueue - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAQueue

```
public:~IAQueue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAQueue

Constructs an IAQueue object.

Overload 1

```
public:IAQueue(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IAQueue(IAQueue < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAQueue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparision

Use these member functions to compare a collection with a given collection.

compare

```
public:ioc::numeric_t
  compare( IAQueue < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these members to perform miscellaneous functions.

addAsLast

Adds the element to the collection as the last element in sequential order. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- ICursorInvalidException
- IOutOfMemory
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAsLast(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsLast(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

dequeue

Copies the first element of the collection to the given element, and removes it from the collection.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If the element is removed, collection classes supporting notifications send a modifyId notification.

Exception
- IEmptyException

Overload 1

```
public:virtual void dequeue(Element&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void dequeue()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

enqueue

Adds the element to the collection, and sets the cursor to the added element. For ordinary queues, the given element is added as the last element. For priority queues, the element is added at a position determined by the ordering relation provided for the element or key type.

Preconditions

- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection except the given cursor become undefined.
- If the element is added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void enqueue(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void enqueue(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

removeFirst

```
public:virtual void removeFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the first element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects
- All cursors of this collection become undefined.
- If an element was removed, collection classes supporting notifications send a removeId notification.

Exception
- IEmptyException

IAQueue - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IARelation

A **relation** is an unordered collection of zero or more elements that have a key. Element equality is supported, and the values of the elements are relevant.

The keys of the elements are not unique. You can add an element whether or not there is already an element in the collection with the same key.

An example of using a relation is a program that maintains a list of all your relatives, with an individual's relationship to you as the key. You can add an aunt, uncle, grandmother, daughter, father-in-law, and so on. You can add an aunt even if an aunt is already in the collection, because you can have several relatives who have the same relationship to you. (For unique relationships such as mother or father, your program would have to check the collection to make sure it did not already contain a family member with that key, before adding the family member.) You can locate a member of the family, but the family members are not in any particular order.

Class header file: iarel.h

IARelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsDilTable
IGRelationAsHshTable
IRRelation
IGRelationAsList
IGRelation
IGRelationAsTable

IARelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct IARelation objects.

~IARelation

```
public:~IARelation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IARelation

Constructs an IARelation object.

Overload 1

```
public:IARelation(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IARelation(IARelation < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IARelation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Add Elements

Add elements to this collection.

addDifference

```
public:virtual void
  addDifference( IARelation < Element,
                 Key > const&,
                 IARelation < Element,
                 Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IABag::differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IARelation < Element,
                   Key > const&,
                   IARelation < Element,
                   Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IABag::intersectionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.

- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IARelation < Element,
            Key > const&,
            IARelation < Element,
            Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IABag::unionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException., if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

Equality Operators

Compare IARelation objects.

operator !=

```
public:bool operator !=(IARelation < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see ICursor::operator ==.

operator ==

```
public:bool operator ==(IARelation < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Condition
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

Logical Operations

Perform difference, intersection, and union operations on this IARelation object and another one, leaving the results in this object.

differenceWith

```
public:virtual void
  differenceWith( IARelation < Element,
                  Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IARelation < Element,
                    Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects
- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

unionWith

```
public:virtual void unionWith(IARelation < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

Remove Elements

Remove elements from the collection.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in IACollection::removeAt.

Side Effects
- If any elements were removed, all cursors of this collection become undefined.

- If any elements were removed, collection classes supporting notifications send a modifyId notification.

Searching

Find elements in the collection that match a particular value.

locateNext

```
protected:bool locateNext(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection that is equal to the given element, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of the given element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to IACollection::locateNext, each occurrence of an element will be visited exactly once in iteration order.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception

- ICursorInvalidException

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

IARelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityKeyCollection`	889		

Inherited Protected Data

None

IARestrictedAccessCollection

IARestrictedAccessCollection is an abstract class. It cannot be used to create any objects. IARestrictedAccessCollection defines the interfaces for the restricted access collections stack, queue, and deque.

Class header file: iarstrct.h

IARestrictedAccessCollection - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IAStack

IAPriorityQueue

IADeque

IAQueue

IARestrictedAccessCollection - Member Functions and Data by Group

Constructors & Destructor

IARestrictedAccessCollection is an abstract class, therefore you cannot create or destroy any objects of this class.

~IARestrictedAccessCollection

```
public:virtual ~IARestrictedAccessCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IARestrictedAccessCollection

Constructs an IARestrictedAccessCollection.

Overload 1

```
public:IARestrictedAccessCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IARestrictedAccessCollection( IARestrictedAccessCollection < Element > c
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IARestrictedAccessCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Add or Remove Elements

Member functions used to add elements to the collection.

add

If the collection is unique (with respect to elements or keys) and the element or key is already contained in the collection, sets the cursor to the existing element in the collection without adding the element. Otherwise, it adds the element to the collection and sets the cursor to the added element. In sequential collections, the given element is added as the last element. In sorted collections, the element is added at a position determined by the element or key value. Adding an element will either use the element's copy constructor or the assignment operator provided for the element type, depending on the implementation variant you choose. See IAEqualityCollection::contains for the definition of element or key containment.

Preconditions

- The cursor must belong to the collection.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If an element was added, all cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Return Value

Returns true if the element was added.

Exceptions

- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

Overload 1

```
public:virtual bool add(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual bool add(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

addAllFrom

```
public:virtual void
  addAllFrom( IARestrictedAccessCollection < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds (copies) all elements of the given collection to the collection. The elements are added in the iteration order of the given collection. The

contents of the elements, not the pointers to the elements, are copied. The elements are added according to the definition of add for this collection. The given collection is not changed.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual add operation:
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send an modifyId notification.

Exceptions
- IOutOfMemory
- IIdenticalCollectionException
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

removeAll

```
public:virtual INumber removeAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Element destructors are called as described in IACollection::removeAt.

Side Effects
- All cursors of this collection become undefined.
- Collection classes supporting notifications send a modifyId notification.

Return Value

The number of elements removed.

Collection Status

Member functions used to obtain status information about the collection.

isBounded

```
public:bool isBounded() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is bounded.

isConsistent

```
public:bool isConsistent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

For internal use only.

isEmpty

```
public:bool isEmpty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is empty.

isFull

```
public:bool isFull() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is bounded and contains the maximum number of elements; that is, if (numberOfElements() == maxNumberOfElements()).

maxNumberOfElements

```
public:INumber maxNumberOfElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the maximum number of elements the collection can contain.

Precondition

The collection is bounded.

Exceptions

- INotBoundedException

numberOfElements

```
public:INumber numberOfElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection.

Copy Elements

Member functions used to copy elements from another collection.

copy

```
public:virtual void
  copy( IARestrictedAccessCollection < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the given collection to this collection. copy() removes all elements from this collection, and adds the elements from the given collection. For information on how adding is done, see IACollection::addAllFrom.

Note: The given collection may be of a concrete type other than the collection itself. In this case, copying implicitly performs a conversion. If, for example, the given collection is a bag and the collection itself is a set, elements with multiple occurrences in the copied bag will only occur once in the resulting set.

Preconditions

Because the elements are copied one by one, the following preconditions are tested for each individual copy operation:

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- All cursors of this collection become undefined.
- If any elements were copied, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection has unique keys. This exception may be thrown, for example, when copying a bag into a map.

Cursor Support

Member functions used to position cursors at particular elements or to create new cursors for the collection.

newCursor

```
public:virtual ICursor* newCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a cursor for the collection and returns a pointer to the cursor. The cursor is initially not valid.

Exception

- IOutOfMemory

newMngCursor

```
public:ICursor* newMngCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a managed cursor for the collection and returns a pointer to the cursor. The cursor is initially not valid.

Exception

- IOutOfMemory

position

```
public:IPosition position(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as positionAt().

positionAt

```
public:IPosition positionAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Determines the position of the current element. position 1 specifies the position of the first element.

Preconditions The cursor must belong to the collection, and the cursor must point to an element of the collection.

Exceptions
- ICursorInvalidException

setToFirst

```
public:bool setToFirst(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first element of the collection in iteration order. If the collection is empty (if no first element exists), it invalidates the given cursor.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if the collection is not empty.

Exception
- ICursorInvalidException

setToLast

```
public:bool setToLast(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last element of the collection in iteration order. If the collection is empty (if no last element exists), the given cursor is no longer valid.

Precondition

The cursor must belong to the collection.

Return Value

Returns true if the collection is not empty.

Exceptions
- ICursorInvalidException

setToNext

```
public:bool setToNext(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order. If no more elements are left to be visited, the given cursor will no longer be valid.

Precondition

The cursor must belong to the collection and must point to an element.

Return Value

Returns true if there is a next element.

Exception
- ICursorInvalidException

setToPosition

```
public:void setToPosition(IPosition, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the element at the given position. Position 1 specifies the first element.

Preconditions
- The cursor must belong to the collection.
- Position must be a valid position in the collection; that is, (1 <= **position** <= numberOfElements()).

Exceptions
- ICursorInvalidException
- IPositionInvalidException

setToPrevious

```
public:bool setToPrevious(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous element in iteration order, or invalidates the cursor if no such element exists.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a previous element exists.

Exception
- ICursorInvalidException

Do For All Elements

Member functions that apply actions to all the elements of the collection.

allElementsDo

Overload 1

```
public:bool
  allElementsDo( bool ( * applicatorFunction ) ( Element const&,
                 void * ) , void* additionalArgument = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements in the collection until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using additionalArgument. The additional argument defaults to zero if no additional argument is given.

Notes:

1. The given function must not remove elements from or add them to the collection. If you want to remove elements, you can use the removeAll() function with a property argument. For further information see removeAll.
2. For the non-const version of allElementsDo(), the given function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the given function returns true for every element it is applied to. Returns false if the collection is empty or if the given function returns false for at least one element it is applied to.

Overload 2

```
public:bool allElementsDo(IConstantApplicator < Element >&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the collection until the applyTo() function returns false. The elements are visited in iteration order. Additional arguments may be passed as arguments to the constructor of the derived applicator class.

Notes:

1. The applyTo() function must not remove elements from or add elements to the collection. If you want to remove elements, you can use the removeAll() function with a property argument.
2. For the non-const version of allElementsDo(), the applyTo() function must not manipulate the element in the collection in a way that changes the positioning property of the element.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to. Returns false if the collection is empty or if the applyTo() function returns false for at least one element it is applied to.

Element Access

Member functions used to elements of the collection.

any

```
public:Element const& any() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as anyElement().

anyElement

```
public:Element const& anyElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to an arbitrary element of the collection.

Precondition

The collection must not be empty.

Exception

- IEmptyException

elementAt

```
public:Element const& elementAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the element pointed to by the given cursor.

Note: For the version of IACollection::elementAt **without** the **const** suffix, do not manipulate the element or the key of the element in the collection in a way that changes the positioning property of the element.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Exception

- ICursorInvalidException

elementAtPosition

```
public:Element const& elementAtPosition(IPosition) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the element at the given position in the collection. Position 1 specifies the first element.

Precondition

(1 <= **position** <= numberOfElements()).

Exception

- IPositionInvalidException

Element Position Testing

Member functions used to test whether elements are at particular positions.

isFirst

```
public:bool isFirst(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as isFirstAt().

isFirstAt

```
public:bool isFirstAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given cursor points to the first element of the collection.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Exception
- ICursorInvalidException

isLast

```
public:bool isLast(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as isLastAt().

isLastAt

```
public:bool isLastAt(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given cursor points to the last element of the collection.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Exception
- ICursorInvalidException

First and Last Element Access

Member functions used to access the first and last elements of the collection.

first

```
public:Element const& first() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as firstElement().

firstElement

```
public:Element const& firstElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the first element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

last

```
public:Element const& last() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Provides the same function as lastElement().

lastElement

```
public:Element const& lastElement() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the last element of the collection.

Precondition

The collection must not be empty.

Exception

- IEmptyException

Notification Support

Member functions used to manage the notification capabilities of the collection.

disableNotification

```
public:INotifier& disableNotification()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stops the notifier from sending notifications to its observers.

enableNotification

```
public:INotifier& enableNotification(bool = true)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Starts the notifier sending notifications to its observers. This function can be overridden by derived classes to perform customized notification that your application might need. For instance, one of your function methods may require that a database be accessible before processing a retrieve function.

isEnabledForNotification

```
public:bool isEnabledForNotification() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the collection is enabled for notification.

notifier

Returns a reference to the notifier object.

Overload 1

```
public:INotifier const& notifier() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:INotifier& notifier()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

notifyObservers

```
public:INotifier& notifyObservers(INotificationEvent const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Notifies all observers in a notifier's list of observers. Each observer receives a notification event containing the identity of the notifier, the notification ID, and any optional data provided by the specific notifier object.

Note:

A public and a protected version of notifyObservers are provided for convenience. The protected version does not require the caller to construct an INotificationEvent to call it. In this case, the construction of the INotificationEvent occurs in the code of the protected notifyObservers function.

operator INotifier &

```
public:operator INotifier &() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the notifier object for this collection.

Streaming

Member functions used to stream elements to and from data streams.

adoptStreamer

```
public:void adoptStreamer(IACollectionStreamer*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adopts a streamer object which will handle streaming the elements to and from a data stream.

operator <<=

```
public:virtual void operator <<=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams the stored elements in from the passed data stream into this collection. It uses the adopted streamer object to do the element streaming work.

operator >>=

```
public:virtual void operator >>=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams the elements of this collection out to the passed data stream. The adopted streamer object is used to do the element streaming.

IARestrictedAccessCollection - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	978		

Inherited Protected Data

None

IASequence

A **sequence** is an ordered collection of elements. The elements are arranged so that each collection has a first and a last element, each element except the last has a next element, and each element but the first has a previous element.

The type and value of the elements are irrelevant, and have no effect on the behavior of the collection. Elements can be added and deleted from any position in the collection. Elements can be retrieved or replaced. A sequence does not support element equality or a key. If you require element equality for a sequence, you can use an equality sequence.

An example of a sequence is a program that maintains a list of the words in a paragraph. The order of the words is obviously important, and you can add or remove words at a given position, but you cannot search for individual words except by iterating through the collection and comparing each word to the word you are searching for. You can add a word that is already present in the sequence, because a given word may be used more than once in a paragraph.

Class header file: iaseq.h

IASequence - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAOrderedCollection
      - IASequentialCollection
        - **IASequence**
          - IGSequenceAsList
          - IGSequenceAsDilTable
          - IRSequence
          - IGSequenceAsTable
          - IGSequence

IASequence - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IASequence

```
public:~IASequence()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASequence

Constructs an IASequence object.

Overload 1

```
public:IASequence(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IASequence(IASequence < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASequence()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions to compare two elements in a given sequence.

compare

```
public:ioc::numeric_t
  compare( IASequence < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

IASequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

Inherited Protected Data

None

IASequentialCollection

IASequentialCollection is an abstract class. It cannot be used to create any objects. The IASequentialCollection class inherits from IAOrderedCollection and defines the interfaces for the properties of ordered elements.

The following concrete classes are defined by IASequentialCollection:
- IEqualitySequence and its variants
- ISequence and its variants

Class header file: iasqntl.h

IASequentialCollection - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAOrderedCollection

IASequentialCollection

IASequence

IASequentialCollection - Member Functions and Data by Group

Constructors & Destructor

These members construct and destruct objects of this class.

~IASequentialCollection

```
public:~IASequentialCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASequentialCollection

Constructs an IASequentialCollection object.

Overload 1

```
public:IASequentialCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IASequentialCollection( IASequentialCollection < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASequentialCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Add Elements

The member functions add elements to the collection.

addAsFirst

Adds the element to the collection as the first element in sequential order. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- ICursorInvalidException
- IOutOfMemory
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAsFirst(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsFirst(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

addAsLast

Adds the element to the collection as the last element in sequential order. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects

- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- ICursorInvalidException
- IOutOfMemory
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAsLast(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsLast(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

addAsNext

```
public:virtual void addAsNext(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the element to the collection as the element following element pointed to by the cursor. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection and must point to an element of the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

addAsPrevious

```
public:virtual void addAsPrevious(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the element to the collection as the element preceding the element pointed to by the cursor. Sets the cursor to the added element.

Preconditions
- The cursor must belong to the collection and must point to an element of the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects

- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IFullException, if the collection is bounded.

addAtPosition

Adds the element at the given position to the collection, and sets the cursor to the added element. If an element exists at the given position, the new element is added as the element preceding the existing element.

Preconditions
- The cursor must belong to the collection.
- (1 <= **position** <= numberOfElements + 1).
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects
- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions
- IOutOfMemory
- ICursorInvalidException
- IPositionInvalidException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAtPosition(IPosition, Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void
  addAtPosition( IPosition,
                 Element const&,
                 ICursor& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Element Access

These member functions provide access to specific elements of the collection.

any

Returns a reference to an arbitrary element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

Overload 1

```
public:Element& any()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element const& any() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

elementAtPosition

Returns a reference to the element at the given position in the collection.

Position 1 specifies the first element.

Position must be a valid position in the collection; that is, (1 <= **position** <= numberOfElements()).

Precondition

(1 <= **position** <= numberOfElements()).

Exception
- IPositionInvalidException

Overload 1

```
public:Element& elementAtPosition(IPosition)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element const& elementAtPosition(IPosition) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

first

Returns a reference to the first element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

Overload 1

```
public:Element& first()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element const& first() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

last

Returns a reference to the last element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

Overload 1

```
public:Element& last()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element const& last() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Element Ordering

These member functions control ordering of the elements.

reverse

```
public:virtual void reverse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reverses the sequence of elements in a collection.

Side Effects

- All cursors of this collection become undefined.
- Collection classes supporting notifications send an modifyId notification.

sort

```
public:virtual void
  sort( ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
        Element const & ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sorts the collection so that the elements occur in ascending order. The relation of two elements is defined by the **comparisonFunction**, which you provide.

Note: The **comparisonFunction** must deliver a result according to the following rules:

>0 if (element1 > element2)
0 if (element1 == element2)
<0 if (element1 < element2)

Side Effects

- All cursors of this collection become undefined.
- Collection classes supporting notifications send a modifyId notification.

IASequentialCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

Inherited Protected Data

None

IASet

A **set** is an ordered collection of zero or more elements with no key. Element equality is supported, and the values of the elements are relevant.

Only unique elements are supported. A request to add an element that already exists is ignored.

An example of a set is a program that creates a packing list for a box of free samples to be sent to a warehouse customer. The program searches a database of in-stock merchandise, and selects ten items at random whose price is below a threshold level. Each item is then added to the set. The set does not allow an item to be added if it is already present in the collection, ensuring that the customer does not get two samples of a single product. The set is not sorted, and elements of the set cannot be located by key.

Class header file: iaset.h

IASet - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection
        IASet
          IGSetAsAvlTree
          IRSet
          IGSetAsList
          IGSetAsDilTable
          IGSet
          IGSetAsBstTree
          IGSetAsTable
          IGSetAsHshTable

IASet - Member Functions and Data by Group

Constructors & Destructor

Use these members to construct and destruct objects of this class.

~IASet

```
public:~IASet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASet

Constructs an IASet object.

Overload 1

```
public:IASet(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IASet(IASet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Miscellaneous

Use these member functions to perform miscellaneous operations.

addDifference

```
public:virtual void
  addDifference( IASet < Element > const&,
                 IASet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IASet::differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExists, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IASet < Element > const&,
                   IASet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IASet::intersectionWith..

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).

- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory:eref.
- IFullException:eref., if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IASet < Element > const&,
            IASet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IASet::unionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

differenceWith

```
public:virtual void differenceWith(IASet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void intersectionWith(IASet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

operator !=

```
public:bool operator !=(IASet < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see IASet::operator ==.

operator ==

```
public:bool operator ==(IASet < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Condition
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.

Type of Collection	Condition
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

unionWith

```
public:virtual void unionWith(IASet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

locateNext

```
protected:bool locateNext(Element const&, ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locates the next element in iteration order in the collection that is equal to the given element, starting at the element next to the one pointed to by the given cursor. Sets the cursor to point to the element in the collection. The cursor is invalidated if the end of the collection is reached and no more occurrences of the given element are left to be visited.

Note: If you code a call to IAEqualitySequence::locateFirst and a set of calls to IAMap::locateNext, each occurrence of an element will be visited exactly once in iteration order.

Precondition

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if an element was found.

Exception
- ICursorInvalidException

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in IACollection::removeAt.

Side Effects
- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

IASet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IASortedBag

A **sorted bag** is an ordered collection of zero or more elements with no key. Both element equality and multiple elements are supported.

An example of using a sorted bag is a program for entering observations on the types of stones found in a riverbed. Each time you find a stone on the riverbed, you enter the stone's mineral type into the collection. You can enter the same mineral type for several stones, because a sorted bag supports multiple elements. You can search for stones of a particular mineral type, and you can determine the number of observations of stones of that type. You can also display the contents of the collection, sorted by mineral type, if you want a complete list of observations made to date.

Class header file: iasb.h

IASortedBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - **IASortedBag**
          - IGSortedBagAsAvlTree
          - IRSortedBag
          - IGSortedBagAsDilTable
          - IGSortedBagAsList
          - IGSortedBag
          - IGSortedBagAsTable
          - IGSortedBagAsBstTree

IASortedBag - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IASortedBag

```
public:~IASortedBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASortedBag

Constructs an IASortedBag object.

Overload 1

```
public:IASortedBag(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IASortedBag(IASortedBag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASortedBag()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these members to compare a collection with a given collection.

compare

```
public:ioc::numeric_t
  compare( IASortedBag < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, IACollection::compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function as described in Using Separate Functions:elink. in the **Class Library User's Guide**.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous functions.

addDifference

```
public:virtual void
  addDifference( IASortedBag < Element > const&,
                 IASortedBag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IASortedBag::differenceWith()

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKEyAlreadyExistsException, if the collection is a map or a sorted map

addIntersection

```
public:virtual void
  addIntersection( IASortedBag < Element > const&,
                   IASortedBag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IASortedBag::intersectionWith().

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyException, if the collection is a map or a sorted map

addUnion

```
public:virtual void
  addUnion( IASortedBag < Element > const&,
            IASortedBag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IASortedBag::unionWith().

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyException, if the collection is a map or a sorted map

differenceWith

```
public:virtual void differenceWith(IASortedBag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IASortedBag < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

numberOfDifferentElements

```
public:INumber numberOfDifferentElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different elements in the collection.

operator !=

```
public:bool operator !=(IASortedBag < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see IACollection::operator ==.

operator ==

```
public:bool operator ==(IASortedBag < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Condition
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

setToNextDifferentElement

```
public:bool setToNextDifferentElement(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception
- ICursorInvalidException

unionWith

```
public:virtual void unionWith(IASortedBag < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyException, if the collection is a map or a sorted map

IASortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualitySortedCollection`	900		

Inherited Protected Data

None

IASortedCollection

IASortedCollection is an abstract class. It cannot be used to create any objects. The IASortedCollection class inherits from IAOrderedCollection:eref and defines the interfaces for the properties of sorted elements.

The following abstract classes are derived from IASortedCollection:
- IAEqualityKeySortedCollection
- IAEqualitySortedCollection
- IAKeySortedCollection

Class header file: iasrt.h

IASortedCollection - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASortedCollection

IASortedCollection - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IASortedCollection

```
public:~IASortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASortedCollection

Constructs an IASortedCollection object.

Overload 1

```
public:IASortedCollection(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IASortedCollection(IASortedCollection < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASortedCollection()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the

required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

IASortedCollection - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	elementAtPosition	951
first	951	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	last	951
lastElement	952	position	953
positionAt	953	removeAtPosition	953
removeFirst	953	removeLast	954
setToLast	950	setToPosition	950
setToPrevious	950		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

Inherited Protected Data

None

IASortedMap

A **sorted map** is an ordered collection of zero or more elements that have a key. Element equality is supported and the values of the elements are relevant. Elements are sorted by the value of their keys.

Only elements with unique keys are supported. A request to add an element whose key already exists in another element of the collection causes an exception to be thrown. A request to add a duplicate element is ignored.

An example of using a sorted map is a program that matches the names of rivers and lakes to their coordinates on a topographical map. The river or lake name is the key. You cannot add a lake or river to the collection if it is already present in the collection. You can display a list of all lakes and rivers, sorted by their names, and you can locate a given lake or river by its key, to determine its coordinates.

Class header file: iasm.h

IASortedMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsAvlTree
IGSortedMapAsList

IGSortedMapAsDilTable
IRSortedMap
IGSortedMapAsBstTree
IGSortedMapAsTable
IGSortedMap

IASortedMap - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

~IASortedMap

```
public:~IASortedMap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASortedMap

Constructs an IASortedMap object.

Overload 1

```
public:IASortedMap(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IASortedMap(IASortedMap < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASortedMap()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions as comparison operators.

compare

```
public:ioc::numeric_t
  compare( IASortedMap < Element,
           Key > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

addDifference

```
public:virtual void
  addDifference( IASortedMap < Element,
                 Key > const&,
                 IASortedMap < Element,
                 Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IASortedMap < Element,
                   Key > const&,
                   IASortedMap < Element,
                   Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see intersectionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IASortedMap < Element,
            Key > const&,
            IASortedMap < Element,
            Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see unionWith().

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

differenceWith

```
public:virtual void
  differenceWith( IASortedMap < Element,
                  Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IASortedMap < Element,
                    Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**)times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

operator !=

```
public:bool operator !=(IASortedMap < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor does not point to the same element (of the same collection) as the given cursor.

operator ==

```
public:bool operator ==(IASortedMap < Element, Key > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to the same element (of the same collection) as the given cursor.

unionWith

```
public:virtual void unionWith(IASortedMap < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

numberOfDifferentKeys

```
protected:INumber numberOfDifferentKeys() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of different keys in the collection.

numberOfElementsWithKey

```
protected:INumber numberOfElementsWithKey(Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements in the collection with the given key.

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

removeAllElementsWithKey

```
protected:INumber removeAllElementsWithKey(Key const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection with the same key as the given key. Element destructors are called as described in removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a removeId notification.

Return Value

The number of elements removed.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

setToNextWithDifferentKey

```
protected:bool setToNextWithDifferentKey(ICursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the collection in iteration order with a key different from the key of the element pointed to by the given cursor. If no such element exists, the given cursor is no longer valid.

Preconditions

The cursor must belong to the collection and must point to an element of the collection.

Return Value

Returns true if a subsequent element was found whose key is different from the current key.

Exception

- ICursorInvalidException

IASortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

Inherited Protected Data

None

IASortedRelation

A **sorted relation** is an ordered collection of zero or more elements that have a key. The lements are sorted by the value of their key. Element equality is supported, and the values of the elements are relevant.

The keys of the elements are not unique. You can add an element whether or not there is already an element in the collection with the same key.

An example of using a sorted relation is a program used by telephone operators to provide directory assistance. The computerized directory is a sorted relation whose key is the name of the individual or business associated with a telephone number. When a caller requests the number of a given person or company, the operator enters the name of that person or company to access the phone number. The collection can have multiple identical keys, because two individuals or companies might have the same name. The collection is sorted alphabetically, because once a year it is used as the source material for a printed telephone directory.

Class header file: iasr.h

IASortedRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsDilTable
IRSortedRelation
IGSortedRelation
IGSortedRelationAsTable

IASortedRelation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IASortedRelation

```
public:~IASortedRelation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASortedRelation

Constructs an IASortedRelation object.

Overload 1

```
public:IASortedRelation(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IASortedRelation(IASortedRelation < Element, Key > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructs a collection.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASortedRelation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions to compare two collections.

compare

```
public:ioc::numeric_t
  compare( IASortedRelation < Element,
           Key > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, IACollection::compare is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only

for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

addDifference

```
public:virtual void
  addDifference( IASortedRelation < Element,
                 Key > const&,
                 IASortedRelation < Element,
                 Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see IASortedRelation::differenceWith

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IASortedRelation < Element,
                   Key > const&,
                   IASortedRelation < Element,
                   Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see IASortedRelation::intersectionWith

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IASortedRelation < Element,
            Key > const&,
            IASortedRelation < Element,
            Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see IASortedRelation::unionWith

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

differenceWith

```
public:virtual void
  differenceWith( IASortedRelation < Element,
                  Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m** > **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IASortedRelation < Element,
                    Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**): times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

operator !=

```
public:bool
  operator !=( IASortedRelation < Element,
               Key > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is not equal to the collection. For a definition of equality for collections, see IACollection::operator==.

operator ==

```
public:bool
  operator ==( IASortedRelation < Element,
               Key > const& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the given collection is equal to the collection. Two collections are equal if the number of elements in each collection is the same, and if the condition for the collection is described in the following list:

Type of Collection	Condition
Unique Elements	If the collections have unique elements, any element that occurs in one collection must occur in the other collection.
Non-Unique Elements	If an element has **n** occurrences in one collection, it must have exactly **n** occurrences in the other collection.
Sequential	The ordering of the elements is the same for both collections.

unionWith

```
public:virtual void
  unionWith( IASortedRelation < Element,
             Key > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in IACollection::removeAt

Side Effects
- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

IASortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityKeySortedCollection`	892		

Inherited Protected Data

None

IASortedSet

A **sorted set** is an ordered collection of zero or more elements with element equality but no key. Only unique elements are supported. A request to add an element that already exists is ignored. The value of the elements is relevant.

The elements of a sorted set are ordered such that the value of each element is less than or equal to the value of its successor.

The element with the smallest value currently in a sorted set is called the **first** element. The element with the largest value is called the **last** element. When an element is added, it is placed in the sorted set according to the defined ordering relation.

An example of using a sorted set is a program that tests numbers to see if they are prime. Two complementary sorted sets are used, one for prime numbers, and one for nonprime numbers. When you enter a number, the program first looks in the set of nonprime numbers. If the value is found there, the number is nonprime. If the value is not found there, the program looks in the set of prime numbers. If the value is found there, the number is prime. Otherwise the program determines whether the number is prime or nonprime, and places it in the appropriate sorted set. The program can also display a list of prime or nonprime numbers, beginning at the first prime or nonprime following a given value, because the numbers in a sorted set are sorted from smallest to largest.

Class header file: iass.h

IASortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IRSortedSet
IGSortedSetAsBstTree
IGSortedSetAsAvlTree
IGSortedSet
IGSortedSetAsDilTable
IGSortedSetAsList
IGSortedSetAsTable

IASortedSet - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

~IASortedSet

```
public:~IASortedSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IASortedSet

Constructs an IASortedSet object.

Overload 1

```
public:IASortedSet(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception

- IOutOfMemory

Overload 2

```
protected:IASortedSet(IASortedSet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IASortedSet()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions as comparison operators.

compare

```
public:ioc::numeric_t
  compare( IASortedSet < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

addDifference

```
public:virtual void
  addDifference( IASortedSet < Element > const&,
                 IASortedSet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the difference between the two given collections, and adds this difference to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the difference between two collections, see differenceWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addIntersection

```
public:virtual void
  addIntersection( IASortedSet < Element > const&,
                   IASortedSet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the intersection of the two given collections, and adds this intersection to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the intersection of two collections, see intersectionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).

- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

addUnion

```
public:virtual void
  addUnion( IASortedSet < Element > const&,
            IASortedSet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates the union of the two given collections, and adds this union to the collection. The contents of the added elements, not the pointers to those elements, are copied.

For a definition of the union of two collections, see unionWith.

Preconditions

Because the elements are added one by one, the following preconditions are tested for each individual addition.
- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects
- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions
- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

differenceWith

```
public:virtual void differenceWith(IASortedSet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the difference between the collection and the given collection. The **difference** of A and B (A minus B) is the set of elements that are contained in A but not in B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **difference** of P and Q contains the element X **m-n** times if **m**> **n**, and zero times if **m**<=**n**.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

intersectionWith

```
public:virtual void
  intersectionWith( IASortedSet < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the intersection of the collection and the given collection. The **intersection** of A and B is the set of elements that is contained in both A and B.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **intersection** of P and Q contains the element X MIN(**m**,**n**)times.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

operator !=

```
public:bool operator !=(IASortedSet < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor does not point to the same element (of the same collection) as the given cursor.

operator ==

```
public:bool operator ==(IASortedSet < Element > const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to the same element (of the same collection) as the given cursor.

unionWith

```
public:virtual void unionWith(IASortedSet < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Makes the collection the union of the collection and the given collection. The **union** of A and B is the set of elements that are members of A or B or both.

The following rule applies for bags with duplicate elements: If bag P contains the element X **m** times and bag Q contains the element X **n** times, the **union** of P and Q contains the element X **m+n** times.

Preconditions

Because the elements from the given collection are added to the collection one by one, the following preconditions are tested for each individual add operation :

- If the collection is bounded and unique, the element or key must exist or (numberOfElements() < maxNumberOfElements()).
- If the collection is bounded and nonunique, (numberOfElements() < maxNumberOfElements()).
- If the collection is a map or a sorted map and contains an element with the same key as the given element, this element must be equal to the given element.

Side Effects

- If any elements were added, all cursors of this collection become undefined.
- If any elements were added, collection classes supporting notifications send a modifyId notification.

Exceptions

- IOutOfMemory
- IFullException, if the collection is bounded
- IKeyAlreadyExistsException, if the collection is a map or a sorted map.

numberOfOccurrences

```
protected:INumber numberOfOccurrences(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of occurrences of the given element in the collection.

removeAllOccurrences

```
protected:INumber removeAllOccurrences(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection that are equal to the given element, and returns the number of elements removed. Element destructors are called as described in removeAt.

Side Effects

- If any elements were removed, all cursors of this collection become undefined.
- If any elements were removed, collection classes supporting notifications send a modifyId notification.

IASortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

Inherited Protected Data

None

IAStack

A **stack** is a sequence with restricted access. It is an unordered collection of elements with no key and no element equality. The elements are arranged so that each collection has a first and a last element, each element except the last has a next element, and each element but the first has a previous element. The type and value of the elements are irrelevant and have no effect on the behavior of the stack.

Elements are added to and deleted from the top of the stack. Consequently, the elements of a stack are in reverse chronological order.

A stack is characterized by a last-in, first-out (LIFO) behavior.

An example of using a stack is a program that keeps track of daily tasks that you have begun to work on but that have been interrupted. When you are working on a task and something else comes up that is more urgent, you enter a description of the interrupted task and where you stopped it into your program, and the task is pushed onto the stack. Whenever you complete a task, you ask the program for the most recently saved task that was interrupted. This task is popped off the stack, and you resume your work where you left off. When you attempt to pop an item off the stack and no item is available, you have completed all your tasks and you can go home.

Class header file: iastk.h

IAStack - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IRStack
IGStackAsList
IGStackAsDilTable
IGStack
IGStackAsTable

IAStack - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IAStack

```
public:~IAStack()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

IAStack

Constructs an IAStack object.

Overload 1

```
public:IAStack(INotifier&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this constructor to create a collection with support for notification. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Overload 2

```
protected:IAStack(IAStack < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.

Constructs a collection and copies all elements from the given collection into the collection as described in the addAllFrom function.

Overload 3

```
protected:IAStack()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is the default constructor. The collection is unbounded and is initially empty.

Note: The collection constructor does not define whether any elements are constructed when the collection is constructed. For some classes, the element's default constructor may be invoked when the collection's constructor is invoked. This happens if a tabular or a diluted sequence implementation variant is used for a collection. The element's default constructor is used to allocate the required storage and initialize the elements. Therefore, a default constructor must be available for elements in such cases.

Exception
- IOutOfMemory

Comparison

Use these member functions to compare a collection with a given collection.

compare

```
public:ioc::numeric_t
  compare( IAStack < Element > const&,
           ioc::numeric_t ( * comparisonFunction ) ( Element const & ,
           Element const & ) ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the collection with the given collection. Comparison yields <0 if the collection is less than the given collection, 0 if the collection is equal to the given collection, and >0 if the collection is greater than the given collection. Comparison is defined by the first pair of corresponding elements, in both collections, that are not equal. If such a pair exists, the collection with the greater element is the greater one. Otherwise, the collection with more elements is the greater one.

Note:

1. The given comparison function must return a result according to the following rules:
   >0 if (element1 > element2)
   0 if (element1 == element2)
   <0 if (element1 < element2)
2. For elements of type char*, compare() is not locale-sensitive by default. Because it uses strcmp() and not strcoll(), it compares the binary values

representing the characters, and is not based on the LC_COLLATE category of the current locale. Its results are reliable only for code pages and character sets in which the collating sequence matches the sequence of binary representations. If you need a comparison based on the LC_COLLATE category, then you must implement your own compare() function.

Return Value

Returns the result of the collection comparison.

Miscellaneous

Use these members to perform miscellaneous functions.

addAsLast

Adds the element to the collection as the last element in sequential order. Sets the cursor to the added element.

Preconditions

- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects

- All cursors of this collection, except the given cursor, become undefined.
- If an element was added, collection classes supporting notifications send an addId notification.

Exceptions

- ICursorInvalidException
- IOutOfMemory
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void addAsLast(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void addAsLast(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pop

Copies the last element of the collection to the given element, and removes it from the collection.

Precondition

The collection must not be empty.

Side Effects

- All cursors of this collection become undefined.
- If the element was removed from the collection, collection classes supporting notifications send a removeId notification.

Exception

- IEmptyException

Overload 1

```
public:virtual void pop()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void pop(Element&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

push

Adds the element to the collection as the last element (as defined for IACollection::add), and sets the cursor to the added element.

Preconditions

- The cursor must belong to the collection.
- If the collection is bounded, (numberOfElements() < maxNumberOfElements()).

Side Effects

- All cursors of this collection, except the given cursor, become undefined.
- If the element was added to the collection, collection classes supporting notifications send an addId notification.

Exceptions

- IOutOfMemory
- IFullException
- IFullException, if the collection is bounded.

Overload 1

```
public:virtual void push(Element const&, ICursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:virtual void push(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

removeLast

```
public:virtual void removeLast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the last element from the collection. Element destructors are called as described in IACollection::removeAt.

Precondition

The collection must not be empty.

Side Effects

All cursors of this collection become undefined.

Exception

- IEmptyException

top

```
public:Element const& top() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the last element of the collection.

Precondition

The collection must not be empty.

Exception
- IEmptyException

IAStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IATree

A **tree** is a collection of **nodes** that can have an arbitrary number of **references** to other nodes. There can be no cycles or short-circuit references. A unique path connects every two nodes. One node is designated as the **root** of the tree.

Formally, a tree can be defined recursively in the following manner:

1. A single **node** by itself is a tree. This node is also the **root** of the tree.
2. If N is a node and T-1, T-2, ..., T-k are trees with roots R-1, R-2, ..., R-k, respectively, then a new tree can be constructed by making N the **parent** of the nodes R-1, R-2, ..., R-k. In this new tree, N is the root and T-1, T-2, ..., T-k are the **subtrees** of the root N. Nodes R-1, R-2, ..., R-k are called **children** of node N.

Associated with each node is a data item called **element**

Nodes without children are called **leaves** or **terminals**. The number of children in a node is called the **degree** of that node. The **level** of a given node is the number of steps in the path from the root to the given node. The root is at level 0 by definition. The **height** of a tree is the length of the longest path from the root to any node.

Class header file: iatree.h

IATree - Hierarchy List

IMetaTree
  IATreeBase
    IATree
      IAMultiwayTree

IATree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

~IATree

```
public:~IATree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this tree.

Side Effects

All cursors of the tree become undefined.

IATree

Constructs an IATree object.

Overload 1

```
protected:IATree(IATree < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree by copying all elements from the given tree.

Exception

- IOutOfMemory

Overload 2

```
protected:IATree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree. The tree is initially empty; that is, it does not contain any nodes.

Add or Remove Elements

Use these member functions to add elements to or remove elements from the tree.

addAsChild

```
public:void
  addAsChild( ITreeCursor const&,
              IPosition,
              Element const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the given element as a child with the given position to the node of this tree denoted by the given cursor.

Preconditions

- The cursor must point to an element of this tree.
- (1 <= **position** <= **numberOfChildren**
- The node denoted by the given cursor (of this tree) must not have a child at the given position.

Exceptions

- IOutOfMemoryException

- ICursorInvalidException
- IPositionInvalidException
- IPositionAlreadyExistsException

addAsRoot

```
public:void addAsRoot(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the given element as root of the tree.

Precondition

The tree must not have a root; that is, it must be empty.

Exceptions

- IOutOfMemoryException.
- IRootAlreadyExistsException

removeAll

```
public:INumber removeAll()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the subtree denoted by the given cursor (of this tree). The IACollection::removeAll function (without a cursor argument) removes all elements from this tree.

Precondition

The cursor must point to an element of this tree.

Exception

- ICursorInvalidException

removeSubtree

```
public:INumber removeSubtree(ITreeCursor&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes the subtree denoted by the given cursor (of this tree). The IACollection::removeAll function (without a cursor argument) removes all elements from this tree.

Precondition

The cursor must point to an element of this tree.

Side Effects

The given cursor is invalidated after removal.

Exception

- ICursorInvalidException

replaceAt

```
public:void replaceAt(ITreeCursor const&, Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Replaces the element pointed to by the cursor with the given element.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

Collection Status

Use these member functions to test the status of the collection.

isConsistent

```
public:bool isConsistent() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal method for testing the consistency of the collection.

isEmpty

```
public:bool isEmpty() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the tree is empty.

numberOfElements

```
public:INumber numberOfElements() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements that the subtree denoted by the given cursor contains. The subtree root, inner, and leaf nodes are counted. The IACollection::numberOfElements function (without a cursor argument) counts the number of elements in the whole tree.

Preconditions

The cursor must belong to the tree and must point to an element in the tree.

Exception
- ICursorInvalidException

numberOfLeaves

```
public:INumber numberOfLeaves() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of leaf elements that the subtree denoted by the given cursor contains. Leaves are nodes that have no children. The IATree::numberOfLeaves function (without a cursor argument) counts the number of leaves in the whole tree.

Preconditions

The cursor must belong to the tree and must point to an element in the tree.

Exception

- ICursorInvalidException

numberOfSubtreeElements

```
public:INumber numberOfSubtreeElements(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of elements that the subtree denoted by the given cursor contains. The subtree root, inner, and leaf nodes are counted. The IACollection::numberOfElements function (without a cursor argument) counts the number of elements in the whole tree.

Preconditions

The cursor must belong to the tree and must point to an element in the tree.

Exception

- ICursorInvalidException

numberOfSubtreeLeaves

```
public:INumber numberOfSubtreeLeaves(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of leaf elements that the subtree denoted by the given cursor contains. Leaves are nodes that have no children. The IATree::numberOfLeaves function (without a cursor argument) counts the number of leaves in the whole tree.

Preconditions

The cursor must belong to the tree and must point to an element in the tree.

Exception

- ICursorInvalidException

Cursor Support

Use these member functions to position a cursor at a particular element in the collection.

newCursor

```
public:ITreeCursor* newCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a cursor for the tree. The cursor is initially invalid.

Return Value

Pointer to the cursor.

Exception

- IOutOfMemory

setToChild

```
public:bool setToChild(IPosition, ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the child with the given position of the node denoted by the given cursor (of this tree). Invalidates the cursor if this child does not exist.

Preconditions
- The cursor must point to an element of this tree.
- (1 <= **position** <= **numberOfChildren**).

Return Value

Returns true if the child exists.

Exceptions
- ICursorInvalidException
- IPositionInvalidException

setToFirst

```
public:bool setToFirst(ITreeCursor&, ITreeIterationOrder) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first node in the given iteration order. Invalidates the cursor if the tree is empty.

Precondition

The cursor must belong to this tree.

Return Value

Returns true if the tree is not empty.

Exception
- ICursorInvalidException

setToFirstExistingChild

```
public:bool setToFirstExistingChild(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first child of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no child. A node with no child is a leaf node of the tree.

Preconditions

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a child.

Exception
- ICursorInvalidException

setToLast

```
public:bool setToLast(ITreeCursor&, ITreeIterationOrder) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last node in the given iteration order. Invalidates the cursor if the tree is empty.

Precondition

The cursor must belong to this tree.

Return Value

Returns true if the tree is not empty.

Exception
- ICursorInvalidException

setToLastExistingChild

```
public:bool setToLastExistingChild(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last child of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no child. A node with no child is a leaf node of the tree.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a child.

Exception
- ICursorInvalidException

setToNext

```
public:bool setToNext(ITreeCursor&, ITreeIterationOrder) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next node in the given iteration order. Invalidates the cursor if there is no next node.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the given cursor does not point to the last node (in iteration order).

Exception
- ICursorInvalidException

setToNextExistingChild

```
public:bool setToNextExistingChild(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next existing sibling of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no next sibling. A node with no next sibling is the last existing child of its parent.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a next sibling.

Exception
- ICursorInvalidException

setToParent

```
public:bool setToParent(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the parent of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no parent. A node with no parent is the root node of its tree.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a parent.

Exception
- ICursorInvalidException

setToPrevious

```
public:bool setToPrevious(ITreeCursor&, ITreeIterationOrder) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous node in the given iteration order. Invalidates the cursor if there is no previous node.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the given cursor does not point to the first node (in iteration order).

Exception
- ICursorInvalidException

setToPreviousExistingChild

```
public:bool setToPreviousExistingChild(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous existing sibling of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no previous sibling. A node with no previous sibling is the first existing child of its parent.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a previous sibling.

Exception
- ICursorInvalidException

setToRoot

```
public:bool setToRoot(ITreeCursor&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the root node of the tree. Invalidates the cursor if the tree is empty (that is, if no root node exists).

Precondition

The cursor must belong to this tree.

Return Value

Returns true if the tree is not empty.

Exception
- ICursorInvalidException

Do For All Elements

These member functions apply actions to all the elements of the tree collection or any subtree of the collection.

allElementsDo

These methods will apply an action to each element in the collection.

Overload 1

```
public:bool
  allElementsDo( IConstantApplicator < Element >&,
                 ITreeIterationOrder ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the subtree denoted by the given cursor (of this tree) until the applyTo() function returns false. The elements are visited in iteration order. The allElementsDo() function (without a subtree cursor argument) iterates over all elements of the tree.

Note: The applyTo() function must not remove elements from or add them to the tree.

Preconditions

- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to.

Exceptions

- ICursorInvalidException.

Overload 2

```
public:bool
  allElementsDo( bool ( * applicatorFunction ) ( Element const&,
                  void * ),
                 ITreeIterationOrder ,
                 void* additionalArgument = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements of the subtree denoted by the given cursor (of this tree) until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Note: The given function must not remove elements from or add them to the tree.

Preconditions

- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the given function returns true for every element it is applied to.

Exceptions

- ICursorInvalidException.

Overload 3

```
public:bool
  allElementsDo( IApplicator < Element >&,
                 ITreeIterationOrder )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the subtree denoted by the given cursor (of this tree) until the applyTo() function returns false. The elements are visited in

iteration order. The allElementsDo() function (without a subtree cursor argument) iterates over all elements of the tree.

Note: The applyTo() function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value Returns true if the applyTo() function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

Overload 4

```
public:bool
  allElementsDo( bool ( * applicatorFunction ) ( Element&,
                  void * ),
                 ITreeIterationOrder ,
                 void* additionalArgument = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements of the subtree denoted by the given cursor (of this tree) until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Note: The given function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the given function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

allSubtreeElementsDo

Applies an action to all of the elements of a particular subtree of the tree.

Overload 1

```
public:bool
  allSubtreeElementsDo( ITreeCursor const&,
                        bool ( * applicatorFunction ) ( Element&,
                         void * ),
                        ITreeIterationOrder ,
                        void* additionalArgument = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements of the subtree denoted by the given cursor (of this tree) until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Note: The given function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the given function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

Overload 2

```
public:bool
  allSubtreeElementsDo( ITreeCursor const&,
                        IConstantApplicator < Element >&,
                        ITreeIterationOrder ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the subtree denoted by the given cursor (of this tree) until the apply To() function returns false. The elements are visited in iteration order.

Note: The applyTo() function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

Overload 3

```
public:bool
  allSubtreeElementsDo( ITreeCursor const&,
                        bool ( * applicatorFunction ) ( Element const&,
                         void * ),
                        ITreeIterationOrder ,
                        void* additionalArgument = 0 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the given function for all elements of the subtree denoted by the given cursor (of this tree) until the given function returns false. The elements are visited in iteration order. Additional arguments can be passed to the given function using **additionalArgument**. The additional argument defaults to zero if no additional argument is given.

Note: The given function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the given function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

Overload 4

```
public:bool
  allSubtreeElementsDo( ITreeCursor const&,
                        IApplicator < Element >&,
                        ITreeIterationOrder )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Calls the applyTo() function of the given applicator for all elements of the subtree denoted by the given cursor (of this tree) until the applyTo() function returns false. The elements are visited in iteration order.

Note: The applyTo() function must not remove elements from or add them to the tree.

Preconditions
- The cursor must belong to this tree.
- The cursor must point to an element of this tree.

Return Value

Returns true if the applyTo() function returns true for every element it is applied to.

Exceptions
- ICursorInvalidException.

Element Access

Use these member functions to access the elements of the collection.

elementAt

Returns a reference to the element pointed to by the given cursor.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

Overload 1

```
public:Element const& elementAt(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:Element& elementAt(ITreeCursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Element Testing

Use these member functions to test elements to see if they are leaf or root nodes.

childPositionAt

```
public:IPosition childPositionAt(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the position of the node pointed to by the given cursor as a child with respect to its parent node. The position of the root node is 1.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

isLeaf

```
public:bool isLeaf(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the node pointed to by the given cursor is a leaf node of the tree. A leaf node is a node with no children.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

isRoot

```
public:bool isRoot(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the node pointed to by the given cursor is the root node of the tree.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

position

```
public:IPosition position(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the position of the node pointed to by the given cursor as a child with respect to its parent node. The position of the root node is 1.

Precondition

The cursor must point to an element of this tree.

Exception
- ICursorInvalidException

Streaming Support

Use these member functions to stream elements to and from data streams.

adoptStreamer

```
public:void adoptStreamer(IACollectionStreamer*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This method adopts a streamer object which is used to handle element streaming to and from a data stream.

operator <<=

```
public:void operator <<=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams in the stored elements to this collection. The adopted streamer object is used to do the streaming.

operator >>=

```
public:void operator >>=(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Streams the elements of this collection out to the passed data stream. The adopted streamer object is used to do the element streaming.

IATree - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1054		

Inherited Protected Data

None

IATreeBase

A **tree** is a collection of **nodes** that can have an arbitrary number of **references** to other nodes. There can be no cycles or short-circuit references. A unique path connects every two nodes. One node is designated as the **root** of the tree.

Formally, a tree can be defined recursively in the following manner:

1. A single **node** by itself is a tree. This node is also the **root** of the tree.
2. If N is a node and T-1, T-2, ..., T-k are trees with roots R-1, R-2, ..., R-k, respectively, then a new tree can be constructed by making N the **parent** of the nodes R-1, R-2, ..., R-k. In this new tree, N is the root and T-1, T-2, ..., T-k are the **subtrees** of the root N. Nodes R-1, R-2, ..., R-k are called **children** of node N.

Associated with each node is a data item called **element**

Nodes without children are called **leaves** or **terminals**. The number of children in a node is called the **degree** of that node. The **level** of a given node is the number of steps in the path from the root to the given node. The root is at level 0 by definition. The **height** of a tree is the length of the longest path from the root to any node.

Class header file: ibtree.h

IATreeBase - Hierarchy List

IMetaTree

IATreeBase

IATree

IATreeBase - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the collection.

~IATreeBase

```
protected: ~IATreeBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys the tree object.

IATreeBase

Constructs a base tree object.

Overload 1

```
protected:IATreeBase(IATreeBase const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor.

Overload 2

```
protected:IATreeBase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor.

Miscellaneous

Use these member functions to perform miscellaneous functions.

CrsrImplOf

Returns a reference to the cursor implementation object of the passed tree cursor object.

Overload 1

```
protected:static ITreeCursorImpl& CrsrImplOf(ITreeCursor& cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a non-const reference.

Overload 2

```
protected:static ITreeCursorImpl const&
  CrsrImplOf( ITreeCursor const& cursor )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a const reference.

InterfaceFor

```
protected:static ITreeCursor* InterfaceFor(ITreeCursorImpl* cursor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor implementation object for the tree.

IATreeBase - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IAutoElemPointer

Use objects of the IAutoElemPointer class as automatic element pointers for collections. Automatic element pointers route the operations on the pointers to the referenced elements, and delete the elements they reference when the pointers are destructed. No reference count is kept.

Class header file: iptr.h

IAutoElemPointer - Hierarchy List

IAutoPointer

IAutoElemPointer

IAutoElemPointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct auto element pointer objects

IAutoElemPointer

Constructs an auto element pointer object.

Overload 1

```
public:IAutoElemPointer(Element*, IExplicitInit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type from a given C++ pointer.

Overload 2

```
public:IAutoElemPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type and initializes it with NULL.

Operators

Use these operators to access the element being managed by this object.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object to which the pointer refers.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a C pointer to the object to which the pointer refers.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicitly converts this pointer to a C++ pointer.

IAutoElemPointer - Inherited Member Functions and Data

Inherited Public Functions

IAutoPointer			
Definition	**Page Number**	**Definition**	**Page Number**
~IAutoPointer operator =	1072 1073	IAutoPointer	1072

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IAutoPointer

Use objects of the IAutoPointer class as automatic pointers for collections. Automatic pointers delete the elements they reference when the pointers are destructed. No reference count is kept.

Class header file: iptr.h

IAutoPointer - Hierarchy List

IAutoPointer

IAutoElemPointer

IAutoPointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct auto pointer objects.

~IAutoPointer

```
public:~IAutoPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes the object referenced by the automatic pointer.

IAutoPointer

Constructs an auto pointer object.

Overload 1

```
public:IAutoPointer(Element*, IExplicitInit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type from a given C++ pointer.

Overload 2

```
public:IAutoPointer(IAutoPointer < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a new pointer and initializes it with the given pointer. The given pointer is set to NULL.

Overload 3

```
public:IAutoPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type and initializes it with NULL.

Operators

Use these operators to access the pointer managed by the auto pointer object.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object to which the pointer refers.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a C pointer to the object to which the pointer refers.

operator =

```
public:void operator =(IAutoPointer < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the given pointer to this pointer. The given pointer is set to NULL and the previously referenced element is deleted.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicitly converts this pointer to a C++ pointer.

IAutoPointer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IBag

IBag is the default implementation variant, as a list, of IABag.

The default implementation of the class IBag requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: ibag.h

IBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBag
IBag

IBag - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IBag objects.

IBag

Constructs an IBag object.

Overload 1

```
public:IBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBag < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBag - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1075	os_mark_schema_type	1075

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag	1133	operator =	1134

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag ImplOf	1133 1135	IGBagTypedef	1134

Inherited Protected Data

None

IBagAsAvlTree

IBagAsAvlTree is the AVL tree implementation variant of IABag.

The implementation of the class IBagAsAvlTree requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: ibagavl.h

IBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsAvlTree
IBagAsAvlTree

IBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IBagAsAvlTree

Constructs an IBagAsAvlTree object.

Overload 1

```
public:IBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1078	os_mark_schema_type	1078

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsAvlTree	1137	operator =	1138

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsAvlTree	1137	IGBagAsAvlTreeTypedef	1138
ImplOf	1139		

Inherited Protected Data

None

IBagAsBstTree

Class header file: ibagbst.h

IBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsBstTree
IBagAsBstTree

IBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IBagAsBstTree

Constructs an IBagAsBstTree object.

Overload 1

```
public:IBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1081	os_mark_schema_type	1081

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsBstTree	1143	operator =	1143

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsBstTree ImplOf	1143 1144	IGBagAsBstTreeTypedef	1144

Inherited Protected Data

None

IBagAsDilTable

IBagAsDilTable is the diluted table implementation variant of IABag.

The implementation of the class IBagAsDilTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: ibagdil.h

IBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsDilTable
IBagAsDilTable

IBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IBagAsDilTable

Constructs an IBagAsDilTable object.

Overload 1

```
public:IBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1085	os_mark_schema_type	1085

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable	1148	operator =	1149

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable ImplOf	1148 1150	IGBagAsDilTableTypedef	1149

Inherited Protected Data

None

IBagAsHshTable

IBagAsHshTable is the hash table implementation variant of IABag.

The implementation of the class IBagAsHshTable requires the following element functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment
- Equality test
- Hash function

Class header file: ibaghsh.h

IBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsHshTable
IBagAsHshTable

IBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IBagAsHshTable

Constructs an IBagAsHshTable object.

Overload 1

```
public:IBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsHshTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1088	os_mark_schema_type	1088

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsHshTable	1154	operator =	1154

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsHshTable	1154	IGBagAsHshTableTypedef	1155
ImplOf	1155		

Inherited Protected Data

None

IBagAsList

IBagAsList is the list implementation variant of IABag.

The implementation of the class IBagAsList requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: ibaglst.h

IBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsList
IBagAsList

IBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IBagAsList

Constructs an IBagAsList object.

Overload 1

```
public:IBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1092	os_mark_schema_type	1092

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsList	1159	operator =	1160

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsList` `ImplOf`	1159 1161	`IGBagAsListTypedef`	1160

Inherited Protected Data

None

IBagAsTable

IBagAsTable is the table implementation variant of IABag.

The implementation of the class IBagAsTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: ibagtab.h

IBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsTable
IBagAsTable

IBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IBagAsTable

Constructs an IBagAsTable object.

Overload 1

```
public:IBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IBagAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1095	os_mark_schema_type	1095

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable	1165	operator =	1165

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable	1165	IGBagAsTableTypedef	1165
ImplOf	1166		

Inherited Protected Data

None

ICollectionGuard

The Class ICollectionGuard is used to provide the IOC collection classes with built-in Guard objects to simplify the explicit serialization needed to protect the collection instance in multithreaded processes. Although the collection instance can be guarded, the locking of elements stored within a collection is the responsibility of the user.

The Guard constructor takes the collection object to be locked, and its destructor unlocks the object.

Class header file: iacllct.h

ICollectionGuard - Hierarchy List

ICollectionGuard

ICollectionGuard - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct ICollectionGuard objects.

~ICollectionGuard

```
public:~ICollectionGuard()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor of ICollectionGuard unlocks the collection object.

Guards for a collection must be destructed before the collection itself is destructed.

ICollectionGuard

```
public:ICollectionGuard( IACollection < Element >&,
                  ioc::numeric_t timeout = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The Guard constructor takes the collection object to be locked and an optional timeout value as parameters.

The timeout value is specified in milliseconds. If a lock request cannot be resolved within the specified range of time, an exception is thrown. The timeout value defaults to -1 to indicate an indefinite wait. The value 0 informs the constructor to throw an exception if the lock is not immediately available.

POSIX Considerations

The timeout value is not supported on POSIX platforms.

ICollectionGuard - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IConstantApplicator

The classes IApplicator and IConstantApplicator define the interface for applicator objects. The redefinition of the function applyTo() defines the actions that are performed with the version of allElementsDo() that takes an applicator argument. Iteration stops when applyTo() returns false.

Class header file: iappl.h

IConstantApplicator - Hierarchy List

IConstantApplicator

IConstantApplicator - Member Functions and Data by Group

Apply To

The applyTo function.

applyTo

```
public:virtual bool applyTo(Element const&) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This functions applies a series of specified statements or a function to all elements of a collection for which you use the applicator. For example, myCollection.allElementsDo(myApplicator); causes the code in the applyTo() function that you code for your applicator object myApplicator to be applied to all elements of the collection myCollection.

IConstantApplicator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICursor

Each collection defines its own nested cursor class. All of these cursor classes are derived from one of the following classes:

- IElementCursor
- IOrderedCursor

IOrderedCursor is derived from IElementCursor, and IElementCursor is in turn derived from ICursor. Only cursors of ordered collections are derived from IOrderedCursor. Cursors from unordered collections are derived from IElementCursor, and only know the member functions from IElement and ICursor.

Because the cursor classes are all abstract classes, no objects of type IOrderedCursor, IElementCursor, or ICursor can be declared. You can obtain cursor objects by using the collection member IACollection::newCursor() or by defining a cursor of a specific collection cursor class. The newCursor() member creates a cursor of the collection to which it is applied.

The newCursor() member returns a pointer to the newly created cursor object.

Each cursor object is associated with a collection object. A cursor function merely calls the corresponding function for this collection. For example, cursor.setToFirst() is the same as collection.setToFirst(cursor), where collection is the object associated with cursor.

Class header file: ibcllct.h

ICursor - Hierarchy List

ICursor

IElementCursor

ICursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ICursor objects.

~ICursor

```
public:virtual ~ICursor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from the collection. Destructors are called for all elements contained in the collection and for elements that have been constructed in advance.

Side Effects

All cursors of the collection become undefined.

ICursor

Constructs the cursor and associates it with the given collection. The cursor is initially invalid. The name of the constructor is that of the nested cursor class.

Overload 1

```
protected:ICursor(Implementation*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:ICursor(ICursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one ICursor object to another.

operator =

```
public:void operator =(ICursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one ICursor object to another.

Miscellaneous

Use these member functions to perform miscellaneous tasks.

copy

```
public:void copy(ICursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies the given cursor to this cursor. This cursor now points to where the given cursor points.

Precondition

The given cursor and this cursor must refer to the same collection type. This precondition cannot be checked.

invalidate

```
public:void invalidate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Invalidates the cursor; that is, it no longer points to an element of the associated collection.

isInBetween

```
public:bool isInBetween() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isMngCursor

```
public:bool isMngCursor() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

isValid

```
public:bool isValid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to an element of the associated collection.

operator !=

```
public:bool operator !=(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor does not point to the same element (of the same collection) as the given cursor.

operator ==

```
public:bool operator ==(ICursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to the same element (of the same collection) as the given cursor.

setToFirst

```
public:bool setToFirst()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first element of the associated collection in iteration order. Invalidates the cursor if the collection is empty (if no first element exists).

Return Value

Returns true if the associated collection is not empty.

setToNext

```
public:bool setToNext()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next element in the associated collection in iteration order. Invalidates the cursor if no more elements are left to be visited. Returns true if there was a next element.

Precondition

The cursor must point to an element of the associated collection.

Exception
- ICursorInvalidException

ICursor - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1100		

Inherited Protected Data

None

IDeque

IDeque is the default implementation variant, as a list, of IADeque.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: idqu.h

IDeque - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IADeque
        IGDeque
          IDeque

IDeque - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IDeque

Constructs an IDeque object.

Overload 1

```
public:IDeque(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IDeque(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDeque - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IDeque < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDeque - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDeque - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IDeque			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1103	os_mark_schema_type	1103

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque	1172	operator =	1172

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque	1172	IGDequeTypedef	1173
ImplOf	1173		

Inherited Protected Data

None

IDequeAsDilTable

IDequeAsDilTable is the diluted table implementation variant of IADeque.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: idqudil.h

IDequeAsDilTable - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IADeque
IGDequeAsDilTable
IDequeAsDilTable

IDequeAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IDequeAsDilTable

Constructs an IDequeAsDilTable object.

Overload 1

```
public:IDequeAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IDequeAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IDequeAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1106	os_mark_schema_type	1106

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	operator =	1176

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	IGDequeAsDilTableTypedef	1176
ImplOf	1177		

Inherited Protected Data

None

IDequeAsList

IDequeAsList is the list implementation variant of IADeque.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: idqulst.h

IDequeAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsList
IDequeAsList

IDequeAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IDequeAsList

Constructs an IDequeAsList object.

Overload 1

```
public:IDequeAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IDequeAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IDequeAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsList - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1109	os_mark_schema_type	1109

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList	1181	operator =	1181

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList	1181	IGDequeAsListTypedef	1182
ImplOf	1182		

Inherited Protected Data

None

IDequeAsTable

IDequeAsTable is the table implementation variant of IADeque.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: idqutab.h

IDequeAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsTable
IDequeAsTable

IDequeAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IDequeAsTable

Constructs an IDequeAsTable object.

Overload 1

```
public:IDequeAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IDequeAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IDequeAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDequeAsTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1112	os_mark_schema_type	1112

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable	1186	operator =	1187

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable ImplOf	1186 1188	IGDequeAsTableTypedef	1187

Inherited Protected Data

None

IElementCursor

Cursors from unordered collections are derived from IElementCursor, and only know the member functions from IElementCursor and ICursor.

Because the cursor classes are all abstract classes, no objects of type IOrderedCursor, IElementCursor, or ICursor can be declared. You can obtain cursor objects by using the collection member IACollection::newCursor() or by defining a cursor of a specific collection cursor class. The newCursor() member creates a cursor of the collection to which it is applied.

The newCursor() member returns a pointer to the newly created cursor object.

Each cursor object is associated with a collection object. A cursor function merely calls the corresponding function for this collection. For example, cursor.setToFirst() is the same as collection.setToFirst(cursor), where collection is the object associated with cursor.

Class header file: iacllct.h

IElementCursor - Hierarchy List

ICursor

IElementCursor

IGRelationAsTableCursor
IGMapAsDilTableCursor
IGKeyBagAsDilTableCursor
IGBagAsHshTableCursor
IOrderedCursor
IGSetCursor
IGMapAsListCursor
IGBagAsListCursor
IGBagAsAvlTreeCursor
IGSetAsAvlTreeCursor
IGKeyBagCursor
IGSetAsDilTableCursor
IGBagCursor
IGHeapCursor
IGHeapAsListCursor
IGBagAsDilTableCursor
IGKeySetAsHshTableCursor
IGBagAsTableCursor
IGKeyBagAsListCursor
IGKeySetCursor
IGMapAsHshTableCursor
IGRelationAsDilTableCursor
IGSetAsBstTreeCursor
IGSetAsHshTableCursor
IGMapAsTableCursor
IGHeapAsDilTableCursor
IGKeyBagAsTableCursor
IGBagAsBstTreeCursor
IGHeapAsTableCursor
IGKeySetAsListCursor
IGSetAsListCursor
IGKeySetAsBstTreeCursor
IGRelationAsHshTableCursor
IGMapAsAvlTreeCursor
IGKeyBagAsHshTableCursor
IGMapCursor

IGKeySetAsDilTableCursor
IGSetAsTableCursor
IGKeySetAsAvlTreeCursor
IGRelationCursor
IGKeySetAsTableCursor
IGRelationAsListCursor
IGMapAsBstTreeCursor

IElementCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IElementCursor

```
protected:IElementCursor(Implementation*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a cursor object.

Miscellaneous

Use these member functions to perform operations on the cursor.

element

```
public:Element const& element() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a constant reference to the element of the associated collection to which the cursor points.

Precondition

The cursor must point to an element of the associated collection.

Exception

- ICursorInvalidException

IElementCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

Inherited Protected Data

None

IElementTreeCursor

A cursor class for enumerating the elements of a tree object.

Class header file: iatree.h

IElementTreeCursor - Hierarchy List

ITreeCursor

IElementTreeCursor

IGMultiwayTreeCursor

IElementTreeCursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IElementTreeCursor

```
protected:IElementTreeCursor(Implementation*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the cursor and associates it with the given tree. The cursor is initially invalid.

Miscellaneous

Use these member functions to perform miscellaneous functions.

element

```
public:Element const& element() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the element of the associated tree to which the cursor points.

Preconditions The cursor must point to a node of the associated tree.

Exception

- ICursorInvalidException

IElementTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ITreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ITreeCursor	2039	copy	2042
invalidate	2042	isValid	2042
operator !=	2043	operator =	2043
operator ==	2043	setToChild	2040
setToFirstExistingChild	2040	setToLastExistingChild	2040
setToNextExistingChild	2041	setToParent	2041
setToPreviousExistingChild	2041	setToRoot	2042

Inherited Public Data

None

Inherited Protected Functions

ITreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	2043	ITreeCursor	2039

Inherited Protected Data

None

IElemPointer

Use objects of the IElemPointer class as element pointers for collections. Element pointers route the operations on the pointers to the referenced elements.

Class header file: iptr.h

IElemPointer - Hierarchy List

IElemPointer

IElemPointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct element pointer objects.

IElemPointer

Constructs an IElemPointer object.

Overload 1

```
public:IElemPointer(Element*, IExplicitInit = IINIT)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type from a given C++ pointer.

Overload 2

```
public:IElemPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type and initializes it with NULL.

Operators

Use these methods to access the element being managed by this object.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object to which the pointer refers.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a C pointer to the object to which the pointer refers.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicitly converts this pointer to a C++ pointer.

IElemPointer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEqualitySequence

IEqualitySequence is the default implementation variant, as a list, of IAEqualitySequence.

The default implementation of IEqualitySequence requires the following element functions:

Element Type

- Assignment
- Copy constructor
- Destructor
- Equality test

Class header file: ies.h

IEqualitySequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequence
IEqualitySequence

IEqualitySequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IEqualitySequence

Constructs an IEqualitySequence object.

Overload 1

```
public:IEqualitySequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IEqualitySequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IEqualitySequence < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IEqualitySequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1121	os_mark_schema_type	1121

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequence	1193	operator =	1194

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequence ImplOf	1193 1195	IGEqualitySequenceTypedef	1194

Inherited Protected Data

None

IEqualitySequenceAsDilTable

IEqualitySequenceAsDilTable is the diluted table implementation variant of IAEqualitySequence.

The implementation of the class IEqualitySequenceAsDilTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test

Class header file: iesdil.h

IEqualitySequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsDilTable
IEqualitySequenceAsDilTable

IEqualitySequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IEqualitySequenceAsDilTable

Constructs an IEqualitySequenceAsDilTable object.

Overload 1

```
public:IEqualitySequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IEqualitySequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IEqualitySequenceAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1124	os_mark_schema_type	1124

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	operator =	1198

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	IGEqualitySequenceAsDilTableTypedef	1198
ImplOf	1199		

Inherited Protected Data

None

IEqualitySequenceAsList

IEqualitySequenceAsList is the list implementation variant of IAEqualitySequence.

The implementation of the class IEqualitySequenceAsList requires the following element functions:

Element Type
- Assignment
- Copy constructor
- Destructor
- Equality test

Class header file: ieslst.h

IEqualitySequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsList
IEqualitySequenceAsList

IEqualitySequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IEqualitySequenceAsList

Constructs an IEqualitySequenceAsList object.

Overload 1

```
public:IEqualitySequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IEqualitySequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IEqualitySequenceAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1127	os_mark_schema_type	1127

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList	1203	operator =	1204

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList ImplOf	1203 1205	IGEqualitySequenceAsListTypedef	1204

Inherited Protected Data

None

IEqualitySequenceAsTable

IEqualitySequenceAsTable is the table implementation variant of IAEqualitySequence.

The implementation of the class IEqualitySequenceAsTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test

Class header file: iestab.h

IEqualitySequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsTable
IEqualitySequenceAsTable

IEqualitySequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IEqualitySequenceAsTable

Constructs an IEqualitySequenceAsTable object.

Overload 1

```
public:IEqualitySequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IEqualitySequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IEqualitySequenceAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEqualitySequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1131	os_mark_schema_type	1131

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable	1209	operator =	1209

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable ImplOf	1209 1210	IGEqualitySequenceAsTableTypedef	1210

Inherited Protected Data

None

IGBag

Variant of IBag with generic operations class.

Class header file: ibag.h

IGBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IABag
        - **IGBag**
          - IVGBag
          - IBag

IGBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBag

Constructs an IGBag object.

Overload 1

```
public:IGBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGBag(IGBag < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

Use this operator to assign one IGBag object to another.

operator =

```
public:IGBag < Element , ElementOps >&
  operator =( IGBag < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBag object to another.

Miscellaneous Members

IGBagTypedef

```
protected:IGBagTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBag - Type Definitions

Cursor

```
typedef IGBagCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBag < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBag < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBag - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1135	os_mark_schema_type	1135

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1135		

Inherited Protected Data

None

IGBagAsAvlTree

Class header file: ibagavl.h

IGBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsAvlTree
IBagAsAvlTree
IVGBagAsAvlTree

IGBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsAvlTree

Constructs an IGBagAsAvlTree object.

Overload 1

```
public:IGBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGBagAsAvlTree( IGBagAsAvlTree < Element,
                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsAvlTree object to another.

operator =

```
public:IGBagAsAvlTree < Element , ElementOps >&
  operator =( IGBagAsAvlTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsAvlTree object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGBagAsAvlTreeTypedef

```
protected:IGBagAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsAvlTree - Type Definitions

Cursor

```
typedef IGBagAsAvlTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsAvlTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsAvlTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsAvlTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1139	os_mark_schema_type	1139

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1139		

Inherited Protected Data

None

IGBagAsAvlTreeCursor

Class header file: ibagavl.h

IGBagAsAvlTreeCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsAvlTreeCursor

IGBagAsAvlTreeMngCursor

IGBagAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsAvlTreeCursor

Constructs an IGBagAsAvlTreeCursor object.

Overload 1

```
public:IGBagAsAvlTreeCursor( IGBagAsAvlTree < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsAvlTreeCursor object to another.

operator =

```
public:IGBagAsAvlTreeCursor < Element , ElementOps >&
  operator =( IGBagAsAvlTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsAvlTreeCursor object to another.

IGBagAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGBagAsBstTree

Class header file: ibagbst.h

IGBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsBstTree
IBagAsBstTree
IVGBagAsBstTree

IGBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGBagAsBstTree

Constructs an IGBagAsBstTree object.

Overload 1

```
public:IGBagAsBstTree( IGBagAsBstTree < Element,
                       ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsBstTree object to another.
operator =

```
public:IGBagAsBstTree < Element , ElementOps >&
  operator =( IGBagAsBstTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsBstTree object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGBagAsBstTreeTypedef

```
protected:IGBagAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsBstTree - Type Definitions

Cursor

```
typedef IGBagAsBstTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsBstTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsBstTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsBstTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1144	os_mark_schema_type	1144

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1144		

Inherited Protected Data

None

IGBagAsBstTreeCursor

Class header file: ibagbst.h

IGBagAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsBstTreeCursor

IGBagAsBstTreeMngCursor

IGBagAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsBstTreeCursor

Constructs an IGBagAsBstTreeCursor object.

Overload 1

```
public:IGBagAsBstTreeCursor( IGBagAsBstTree < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsBstTreeCursor object to another.

operator =

```
public:IGBagAsBstTreeCursor < Element , ElementOps >&
  operator =( IGBagAsBstTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsBstTreeCursor object to another.

IGBagAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`element`	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGBagAsDilTable

Variant of IBagAsDilTable with generic operations class.

Class header file: ibagdil.h

IGBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsDilTable
IVGBagAsDilTable
IBagAsDilTable

IGBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsDilTable

Constructs an IGBagAsDilTable object.

Overload 1

```
public:IGBagAsDilTable( IGBagAsDilTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsDilTable object to another.

operator =

```
public:IGBagAsDilTable < Element , ElementOps >&
  operator =( IGBagAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsDilTable object to another.

Miscellaneous Members

IGBagAsDilTableTypedef

```
protected:IGBagAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsDilTable - Type Definitions

Cursor

```
typedef IGBagAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1150	os_mark_schema_type	1150

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1150		

Inherited Protected Data

None

IGBagAsDilTableCursor

Class header file: ibagdil.h

IGBagAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsDilTableCursor

IGBagAsDilTableMngCursor

IGBagAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsDilTableCursor

Constructs an IGBagAsDilTableCursor object.

Overload 1

```
public:IGBagAsDilTableCursor( IGBagAsDilTable < Element,
                              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsDilTableCursor object to another.

operator =

```
public:IGBagAsDilTableCursor < Element , ElementOps >&
  operator =( IGBagAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsAvlTreeCursor object to another.

IGBagAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGBagAsHshTable

Variant of IBagAsHshTable with generic operations class.

Class header file: ibaghsh.h

IGBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsHshTable
IBagAsHshTable
IVGBagAsHshTable

IGBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsHshTable

Constructs an IGBagAsHshTable object.

Overload 1

```
public:IGBagAsHshTable( IGBagAsHshTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsHshTable object to another.

operator =

```
public:IGBagAsHshTable < Element , ElementOps >&
  operator =( IGBagAsHshTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsHshTree object to another.

Miscellaneous Members

IGBagAsHshTableTypedef

```
protected:IGBagAsHshTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsHshTable - Type Definitions

Cursor

```
typedef IGBagAsHshTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsHshTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsHshTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsHshTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1155	os_mark_schema_type	1155

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1155		

Inherited Protected Data

None

IGBagAsHshTableCursor

Class header file: ibaghsh.h

IGBagAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsHshTableCursor

IGBagAsHshTableMngCursor

IGBagAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsHshTableCursor

Constructs an IGBagAsHshTableCursor object.

Overload 1

```
public:IGBagAsHshTableCursor( IGBagAsHshTable < Element,
                              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsHshTableCursor object to another.

operator =

```
public:IGBagAsHshTableCursor < Element , ElementOps >&
  operator =( IGBagAsHshTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsHshTableCursor object to another.

IGBagAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGBagAsList

Variant of IBagAsList with generic operations class.

Class header file: ibaglst.h

IGBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsList
IVGBagAsList
IBagAsList

IGBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsList

Constructs an IGBagAsList object.

Overload 1

```
public:IGBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGBagAsList(IGBagAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsList object to another.

operator =

```
public:IGBagAsList < Element , ElementOps >&
  operator =( IGBagAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsList object to another.

Miscellaneous Members

IGBagAsListTypedef

```
protected:IGBagAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsList - Type Definitions

Cursor

```
typedef IGBagAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1161	os_mark_schema_type	1161

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1161		

Inherited Protected Data

None

IGBagAsListCursor

Class header file: ibaglst.h

IGBagAsListCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsListCursor

IGBagAsListMngCursor

IGBagAsListCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsListCursor

Constructs an IGBagAsListCursor object.

Overload 1

```
public:IGBagAsListCursor( IGBagAsList < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsListCursor object to another.

operator =

```
public:IGBagAsListCursor < Element , ElementOps >&
  operator =( IGBagAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsListCursor object to another.

IGBagAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGBagAsTable

Variant of IBagAsTable with generic operations class.

Class header file: ibagtab.h

IGBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsTable
IVGBagAsTable
IBagAsTable

IGBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsTable

Constructs an IGBagAsTable object.

Overload 1

```
public:IGBagAsTable(IGBagAsTable < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsTable object to another.

operator =

```
public:IGBagAsTable < Element , ElementOps >&
  operator =( IGBagAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsTable object to another.

Miscellaneous Members

IGBagAsTableTypedef

```
protected:IGBagAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsTable - Type Definitions

Cursor

```
typedef IGBagAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGBagAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IBagAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGBagAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGBagAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1166	os_mark_schema_type	1166

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1166		

Inherited Protected Data

None

IGBagAsTableCursor

Class header file: ibagtab.h

IGBagAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGBagAsTableCursor

IGBagAsTableMngCursor

IGBagAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGBagAsTableCursor

Constructs an IGBagAsTableCursor object.

Overload 1

```
public:IGBagAsTableCursor( IGBagAsTable < Element,
                           ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsTableCursor object to another.

operator =

```
public:IGBagAsTableCursor < Element , ElementOps >&
  operator =( IGBagAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagAsTableCursor object to another.

IGBagAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`element`	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGBagCursor

Class header file: ibag.h

IGBagCursor - Hierarchy List

ICursor

IElementCursor

IGBagCursor

IGBagMngCursor

IGBagCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct IGBagCursor objects.

IGBagCursor

Constructs an IGBagCursor object.

Overload 1

```
public:IGBagCursor(IGBag < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGBagCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagCursor object to another.

operator =

```
public:IGBagCursor < Element , ElementOps >&
  operator =( IGBagCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGBagCursor object to another.

IGBagCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGDeque

Variant of IDeque with generic operations class.

Class header file: idqu.h

IGDeque - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IADeque
        IGDeque
          IVGDeque
          IDeque

IGDeque - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGDeque

Constructs an IGDeque object.

Overload 1

```
public:IGDeque(IGDeque < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGDeque(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGDeque(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGDeque object to another.

operator =

```
public:IGDeque < Element , ElementOps >&
  operator =( IGDeque < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGDeque object to another.

Miscellaneous Members

IGDequeTypedef

```
protected:IGDequeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDeque - Type Definitions

Cursor

```
typedef IGDequeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGDequeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGDeque < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDeque - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGDeque < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDeque - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1173	os_mark_schema_type	1173

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1173		

Inherited Protected Data

None

IGDequeAsDilTable

Variant of IDequeAsDilTable with generic operations class.

Class header file: idqudil.h

IGDequeAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsDilTable
IDequeAsDilTable
IVGDequeAsDilTable

IGDequeAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGDequeAsDilTable

Constructs an IGDequeAsDilTable object.

Overload 1

```
public:IGDequeAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGDequeAsDilTable( IGDequeAsDilTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGDequeAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGDequeAsDilTable object to another.

operator =

```
public:IGDequeAsDilTable < Element , ElementOps >&
  operator =( IGDequeAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGDequeAsDilTable object to another.

Miscellaneous Members

IGDequeAsDilTableTypedef

```
protected:IGDequeAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsDilTable - Type Definitions

Cursor

```
typedef IGDequeAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGDequeAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGDequeAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGDequeAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1177	os_mark_schema_type	1177

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1177		

Inherited Protected Data

None

IGDequeAsDilTableCursor

Class header file: idqudil.h

IGDequeAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGDequeAsDilTableCursor

IGDequeAsDilTableMngCursor

IGDequeAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGDequeAsDilTableCursor

Overload 1

```
public:IGDequeAsDilTableCursor( IGDequeAsDilTable
                                    < Element,ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGDequeAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGDequeAsDilTableCursor < Element , ElementOps >&
  operator =( IGDequeAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGDequeAsList

Variant of IDequeAsList with generic operations class.

Class header file: idqulst.h

IGDequeAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsList
IVGDequeAsList
IDequeAsList

IGDequeAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGDequeAsList

Constructs an IGDequeAsList object.

Overload 1

```
public:IGDequeAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGDequeAsList(IGDequeAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGDequeAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGDequeAsList object to another.

operator =

```
public:IGDequeAsList < Element , ElementOps >&
  operator =( IGDequeAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGDequeAsList object to another.

Miscellaneous Members

IGDequeAsListTypedef

```
protected:IGDequeAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsList - Type Definitions

Cursor

```
typedef IGDequeAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGDequeAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGDequeAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGDequeAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsList - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1182	os_mark_schema_type	1182

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1182		

Inherited Protected Data

None

IGDequeAsListCursor

Class header file: idqulst.h

IGDequeAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGDequeAsListCursor
        IGDequeAsListMngCursor

IGDequeAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGDequeAsListCursor

Overload 1

```
public:IGDequeAsListCursor( IGDequeAsList < Element,
                            ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGDequeAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGDequeAsListCursor < Element , ElementOps >&
  operator =( IGDequeAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGDequeAsTable

Variant of IDequeAsTable with generic operations class.

Class header file: idqutab.h

IGDequeAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsTable
IDequeAsTable
IVGDequeAsTable

IGDequeAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGDequeAsTable

Constructs an IGDequeAsTable object.

Overload 1

```
public:IGDequeAsTable( IGDequeAsTable < Element,
                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGDequeAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGDequeAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGDequeAsTable object to another.

operator =

```
public:IGDequeAsTable < Element , ElementOps >&
  operator =( IGDequeAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGDequeAsTable object to another.

Miscellaneous Members

IGDequeAsTableTypedef

```
protected:IGDequeAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsTable - Type Definitions

Cursor

```
typedef IGDequeAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGDequeAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGDequeAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGDequeAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1188	os_mark_schema_type	1188

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1188		

Inherited Protected Data

None

IGDequeAsTableCursor

Class header file: idqutab.h

IGDequeAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGDequeAsTableCursor

IGDequeAsTableMngCursor

IGDequeAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGDequeAsTableCursor

Overload 1

```
public:IGDequeAsTableCursor( IGDequeAsTable < Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGDequeAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGDequeAsTableCursor < Element , ElementOps >&
  operator =( IGDequeAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGDequeCursor

Class header file: idqu.h

IGDequeCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGDequeCursor
        IGDequeMngCursor

IGDequeCursor - Member Functions and Data by Group

Constructors & Destructor

IGDequeCursor

Overload 1

```
public:IGDequeCursor(IGDeque < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGDequeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGDequeCursor < Element , ElementOps >&
  operator =( IGDequeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGDequeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGEqualitySequence

Variant of IEqualitySequence with generic operations class.

Class header file: ies.h

IGEqualitySequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequence
IVGEqualitySequence
IEqualitySequence

IGEqualitySequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGEqualitySequence

Constructs an IGEqualitySequence object.

Overload 1

```
public:IGEqualitySequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGEqualitySequence( IGEqualitySequence < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGEqualitySequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGEqualitySequence object to another.

operator =

```
public:IGEqualitySequence < Element , ElementOps >&
  operator =( IGEqualitySequence < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGEqualitySequence object to this object.

Miscellaneous Members

IGEqualitySequenceTypedef

```
protected:IGEqualitySequenceTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequence - Type Definitions

Cursor

```
typedef IGEqualitySequenceCursor < Element ,
        ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

Implementation

This member cannot be accessed.

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGEqualitySequenceMngCursor < Element ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IEqualitySequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

Self

```
typedef IGEqualitySequence < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

IGEqualitySequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGEqualitySequence < Element,
          ElementOps > const& )
```

This member cannot be accessed.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1195	os_mark_schema_type	1195

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1195		

Inherited Protected Data

None

IGEqualitySequenceAsDilTable

Variant of IEqualitySequenceAsDilTable with generic operations class.

Class header file: iesdil.h

IGEqualitySequenceAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySequence
        - **IGEqualitySequenceAsDilTable**
          - IVGEqualitySequenceAsDilTable
          - IEqualitySequenceAsDilTable

IGEqualitySequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGEqualitySequenceAsDilTable

Constructs an IGEqualitySequenceAsDilTable object.

Overload 1

```
public:IGEqualitySequenceAsDilTable( IGEqualitySequenceAsDilTable < Element,
                                     ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGEqualitySequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGEqualitySequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGEqualitySequenceAsDilTable object to another.

operator =

```
public:IGEqualitySequenceAsDilTable < Element , ElementOps >&
  operator =( IGEqualitySequenceAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGEqualitySequenceAsDilTable object to this object.

Miscellaneous Members

IGEqualitySequenceAsDilTableTypedef

```
protected:IGEqualitySequenceAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsDilTable - Type Definitions

Cursor

```
typedef IGEqualitySequenceAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGEqualitySequenceAsDilTableMngCursor < Element ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IEqualitySequenceAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGEqualitySequenceAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGEqualitySequenceAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1199	os_mark_schema_type	1199

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1199		

Inherited Protected Data

None

IGEqualitySequenceAsDilTableCursor

Class header file: iesdil.h

IGEqualitySequenceAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGEqualitySequenceAsDilTableCursor
        IGEqualitySequenceAsDilTableMngCursor

IGEqualitySequenceAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGEqualitySequenceAsDilTableCursor

Overload 1

```
public:IGEqualitySequenceAsDilTableCursor( IGEqualitySequenceAsDilTable < Element,
                                           ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGEqualitySequenceAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGEqualitySequenceAsDilTableCursor < Element , ElementOps >&
  operator =( IGEqualitySequenceAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGEqualitySequenceAsList

Variant of IEqualitySequenceAsList with generic operations class.

Class header file: ieslst.h

IGEqualitySequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsList
IEqualitySequenceAsList
IVGEqualitySequenceAsList

IGEqualitySequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGEqualitySequenceAsList

Constructs an IGEqualitySequenceAsList object.

Overload 1

```
public:IGEqualitySequenceAsList( IGEqualitySequenceAsList < Element,
                                 ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGEqualitySequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGEqualitySequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGEqualitySequenceAsList object to another.

operator =

```
public:IGEqualitySequenceAsList < Element , ElementOps >&
  operator =( IGEqualitySequenceAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGEqualitySequenceAsList object to this object.

Miscellaneous Members

IGEqualitySequenceAsListTypedef

```
protected:IGEqualitySequenceAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsList - Type Definitions

Cursor

```
typedef IGEqualitySequenceAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGEqualitySequenceAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IEqualitySequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGEqualitySequenceAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGEqualitySequenceAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1205	os_mark_schema_type	1205

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1205		

Inherited Protected Data

None

IGEqualitySequenceAsListCursor

Class header file: ieslst.h

IGEqualitySequenceAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGEqualitySequenceAsListCursor

IGEqualitySequenceAsListMngCursor

IGEqualitySequenceAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGEqualitySequenceAsListCursor

Overload 1

```
public:IGEqualitySequenceAsListCursor( IGEqualitySequenceAsList
                                       < Element,ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGEqualitySequenceAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGEqualitySequenceAsListCursor < Element , ElementOps >&
  operator =( IGEqualitySequenceAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGEqualitySequenceAsTable

Variant of IEqualitySequenceAsTable with generic operations class.

Class header file: iestab.h

IGEqualitySequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsTable
IVGEqualitySequenceAsTable
IEqualitySequenceAsTable

IGEqualitySequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGEqualitySequenceAsTable

Constructs an IGEqualitySequenceAsTable object.

Overload 1

```
public:IGEqualitySequenceAsTable( IGEqualitySequenceAsTable < Element,
                                  ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGEqualitySequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGEqualitySequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGEqualitySequenceAsTable object to another.
operator =

```
public:IGEqualitySequenceAsTable < Element , ElementOps >&
  operator =( IGEqualitySequenceAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGEqualitySequenceAsTable object to this object.

Miscellaneous Members

IGEqualitySequenceAsTableTypedef

```
protected:IGEqualitySequenceAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsTable - Type Definitions

Cursor

```
typedef IGEqualitySequenceAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGEqualitySequenceAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IEqualitySequenceAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGEqualitySequenceAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGEqualitySequenceAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1210	os_mark_schema_type	1210

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1210		

Inherited Protected Data

None

IGEqualitySequenceAsTableCursor

Class header file: iestab.h

IGEqualitySequenceAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGEqualitySequenceAsTableCursor

IGEqualitySequenceAsTableMngCursor

IGEqualitySequenceAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGEqualitySequenceAsTableCursor

Overload 1

```
public:IGEqualitySequenceAsTableCursor( IGEqualitySequenceAsTable
                                        < Element,ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGEqualitySequenceAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGEqualitySequenceAsTableCursor < Element , ElementOps >&
  operator =( IGEqualitySequenceAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGEqualitySequenceAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGEqualitySequenceCursor

Class header file: ies.h

IGEqualitySequenceCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGEqualitySequenceCursor

IGEqualitySequenceMngCursor

IGEqualitySequenceCursor - Member Functions and Data by Group

Constructors & Destructor

Construct and destruct objects of this class.

IGEqualitySequenceCursor

Constructs an object of this class.

Overload 1

```
public:IGEqualitySequenceCursor( IGEqualitySequence < Element,
                                 ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGEqualitySequenceCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGEqualitySequenceCursor < Element , ElementOps >&
  operator =( IGEqualitySequenceCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The assignment operator for this class.

IGEqualitySequenceCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGHeap

Variant of IHeap with generic operations class.

Class header file: ihp.h

IGHeap - Hierarchy List

IGHeap

IVGHeap

IHeap

IGHeap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGHeap

Constructs an IGHeap object.

Overload 1

```
public:IGHeap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGHeap(IGHeap < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGHeap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGHeap object to another.

operator =

```
public:IGHeap < Element , ElementOps >&
  operator =( IGHeap < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGHeap object to another.

Miscellaneous Members

IGHeapTypedef

```
protected:IGHeapTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeap - Type Definitions

Cursor

```
typedef IGHeapCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGHeapMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IHeapAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGHeap < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGHeap < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeap - Inherited Member Functions and Data

Inherited Public Functions

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1218	os_mark_schema_type	1218

Inherited Public Data

None

Inherited Protected Functions

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1218		

Inherited Protected Data

None

IGHeapAsDilTable

Variant of IHeapAsDilTable with generic operations class.

Class header file: ihpdil.h

IGHeapAsDilTable - Hierarchy List

IGHeapAsDilTable

IHeapAsDilTable

IVGHeapAsDilTable

IGHeapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGHeapAsDilTable

Constructs an IGHeapAsDilTable object.

Overload 1

```
public:IGHeapAsDilTable( IGHeapAsDilTable < Element,
                         ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGHeapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGHeapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGHeapAsDilTable object to another.

operator =

```
public:IGHeapAsDilTable < Element , ElementOps >&
  operator =( IGHeapAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGHeapAsDilTable object to another.

Miscellaneous Members

IGHeapAsDilTableTypedef

```
protected:IGHeapAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsDilTable - Type Definitions

Cursor

```
typedef IGHeapAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGHeapAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IHeapAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGHeapAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGHeapAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1220	os_mark_schema_type	1220

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1220		

Inherited Protected Data

None

IGHeapAsDilTableCursor

Class header file: ihpdil.h

IGHeapAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGHeapAsDilTableCursor

IGHeapAsDilTableMngCursor

IGHeapAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGHeapAsDilTableCursor

Overload 1

```
public:IGHeapAsDilTableCursor( IGHeapAsDilTable < Element,
                               ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGHeapAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGHeapAsDilTableCursor < Element , ElementOps >&
  operator =( IGHeapAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGHeapAsList

Variant of IHeapAsList with generic operations class.

Class header file: ihplst.h

IGHeapAsList - Hierarchy List

IGHeapAsList

IHeapAsList

IVGHeapAsList

IGHeapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGHeapAsList

Constructs an IGHeapAsList object.

Overload 1

```
public:IGHeapAsList(IGHeapAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGHeapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGHeapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGHeapAsList object to another.

operator =

```
public:IGHeapAsList < Element , ElementOps >&
  operator =( IGHeapAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGHeapAsList object to another.

Miscellaneous Members

IGHeapAsListTypedef

```
protected:IGHeapAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsList - Type Definitions

Cursor

```
typedef IGHeapAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGHeapAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IHeapAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGHeapAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGHeapAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsList - Inherited Member Functions and Data

Inherited Public Functions

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1224	os_mark_schema_type	1224

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1224		

Inherited Protected Data

None

IGHeapAsListCursor

Class header file: ihplst.h

IGHeapAsListCursor - Hierarchy List

ICursor

IElementCursor

IGHeapAsListCursor

IGHeapAsListMngCursor

IGHeapAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGHeapAsListCursor

Overload 1

```
public:IGHeapAsListCursor( IGHeapAsList < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGHeapAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGHeapAsListCursor < Element , ElementOps >&
  operator =( IGHeapAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGHeapAsTable

Variant of IHeapAsTable with generic operations class.

Class header file: ihptab.h

IGHeapAsTable - Hierarchy List

IGHeapAsTable

IVGHeapAsTable

IHeapAsTable

IGHeapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGHeapAsTable

Constructs an IGHeapAsTable object.

Overload 1

```
public:IGHeapAsTable(IGHeapAsTable < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGHeapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGHeapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGHeapAsTable object to another.

operator =

```
public:IGHeapAsTable < Element , ElementOps >&
  operator =( IGHeapAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGHeapAsTable object to another.

Miscellaneous Members

IGHeapAsTableTypedef

```
protected:IGHeapAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsTable - Type Definitions

Cursor

```
typedef IGHeapAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGHeapAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IHeapAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGHeapAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGHeapAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1227	os_mark_schema_type	1228

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1227		

Inherited Protected Data

None

IGHeapAsTableCursor

Class header file: ihptab.h

IGHeapAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGHeapAsTableCursor

IGHeapAsTableMngCursor

IGHeapAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGHeapAsTableCursor

Overload 1

```
public:IGHeapAsTableCursor( IGHeapAsTable < Element,
                            ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGHeapAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGHeapAsTableCursor < Element , ElementOps >&
  operator =( IGHeapAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGHeapCursor

Class header file: ihp.h

IGHeapCursor - Hierarchy List

ICursor

IElementCursor

IGHeapCursor

IGHeapMngCursor

IGHeapCursor - Member Functions and Data by Group

Constructors & Destructor

IGHeapCursor

Overload 1

```
public:IGHeapCursor(IGHeap < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGHeapCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGHeapCursor < Element , ElementOps >&
  operator =( IGHeapCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGHeapCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGKeyBag

Variant of IKeyBag with generic operations class.

Class header file: ikb.h

IGKeyBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBag
IVGKeyBag
IKeyBag

IGKeyBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBag

Constructs an IGKeyBag object.

Overload 1

```
public:IGKeyBag(IGKeyBag < Element, Key, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeyBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeyBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBag object to another.

operator =

```
public:IGKeyBag < Element , Key , ElementOps >&
  operator =( IGKeyBag < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBag object to another.

Miscellaneous Members

IGKeyBagTypedef

```
protected:IGKeyBagTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBag - Type Definitions

Cursor

```
typedef IGKeyBagCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeyBagMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeyBagAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeyBag < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeyBag < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1233	os_mark_schema_type	1233

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1233		

Inherited Protected Data

None

IGKeyBagAsDilTable

Variant of IKeyBagAsDilTable with generic operations class.

Class header file: ikbdil.h

IGKeyBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsDilTable
IVGKeyBagAsDilTable
IKeyBagAsDilTable

IGKeyBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsDilTable

Constructs an IGKeyBagAsDilTable object.

Overload 1

```
public:IGKeyBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeyBagAsDilTable( IGKeyBagAsDilTable < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeyBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsDilTable object to another.

operator =

```
public:IGKeyBagAsDilTable < Element , Key , ElementOps >&
  operator =( IGKeyBagAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsDilTable object to another.

Miscellaneous Members

IGKeyBagAsDilTableTypedef

```
protected:IGKeyBagAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsDilTable - Type Definitions

Cursor

```
typedef IGKeyBagAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeyBagAsDilTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeyBagAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeyBagAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeyBagAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1236	os_mark_schema_type	1237

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1236		

Inherited Protected Data

None

IGKeyBagAsDilTableCursor

Class header file: ikbdil.h

IGKeyBagAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGKeyBagAsDilTableCursor

IGKeyBagAsDilTableMngCursor

IGKeyBagAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeyBagAsDilTableCursor

Overload 1

```
public:IGKeyBagAsDilTableCursor( IGKeyBagAsDilTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeyBagAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeyBagAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGKeyBagAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`element`	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGKeyBagAsHshTable

Variant of IKeyBagAsHshTable with generic operations class.

Class header file: ikbhsh.h

IGKeyBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsHshTable
IVGKeyBagAsHshTable
IKeyBagAsHshTable

IGKeyBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsHshTable

Constructs an IGKeyBagAsHshTable object.

Overload 1

```
public:IGKeyBagAsHshTable( IGKeyBagAsHshTable < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeyBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeyBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsAvlTree object to another.

operator =

```
public:IGKeyBagAsHshTable < Element , Key , ElementOps >&
  operator =( IGKeyBagAsHshTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsHshTable object to another.

Miscellaneous Members

IGKeyBagAsHshTableTypedef

```
protected:IGKeyBagAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsHshTable - Type Definitions

Cursor

```
typedef IGKeyBagAsHshTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeyBagAsHshTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeyBagAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeyBagAsHshTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeyBagAsHshTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1242	os_mark_schema_type	1242

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAKeyCollection`	909		

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1242		

Inherited Protected Data

None

IGKeyBagAsHshTableCursor

Class header file: ikbhsh.h

IGKeyBagAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGKeyBagAsHshTableCursor

IGKeyBagAsHshTableMngCursor

IGKeyBagAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsHshTableCursor

Constructs an IGKeyBagAsHshTableCursor object.

Overload 1

```
public:IGKeyBagAsHshTableCursor( IGKeyBagAsHshTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeyBagAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsHshTableCursor object to another.

operator =

```
public:IGKeyBagAsHshTableCursor < Element , Key , ElementOps >&
  operator =( IGKeyBagAsHshTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsHshTableCursor object to another.

IGKeyBagAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeyBagAsList

Variant of IKeyBagAsList with generic operations class.

Class header file: ikblst.h

IGKeyBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsList
IKeyBagAsList
IVGKeyBagAsList

IGKeyBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsList

Constructs an IGKeyBagAsList object.

Overload 1

```
public:IGKeyBagAsList( IGKeyBagAsList < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeyBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeyBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsList object to another.

operator =

```
public:IGKeyBagAsList < Element , Key , ElementOps >&
  operator =( IGKeyBagAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsList object to another.

Miscellaneous Members

IGKeyBagAsListTypedef

```
protected:IGKeyBagAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsList - Type Definitions

Cursor

```
typedef IGKeyBagAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeyBagAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeyBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeyBagAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeyBagAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1247	os_mark_schema_type	1247

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1247		

Inherited Protected Data

None

IGKeyBagAsListCursor

Class header file: ikblst.h

IGKeyBagAsListCursor - Hierarchy List

ICursor

IElementCursor

IGKeyBagAsListCursor

IGKeyBagAsListMngCursor

IGKeyBagAsListCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsListCursor

Constructs an IGKeyBagAsListCursor object.

Overload 1

```
public:IGKeyBagAsListCursor( IGKeyBagAsList < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeyBagAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsListCursor object to another.

operator =

```
public:IGKeyBagAsListCursor < Element , Key , ElementOps >&
  operator =( IGKeyBagAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsListCursor object to another.

IGKeyBagAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeyBagAsTable

Variant of IKeyBagAsTable with generic operations class.

Class header file: ikbtab.h

IGKeyBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsTable
IKeyBagAsTable
IVGKeyBagAsTable

IGKeyBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsTable

Constructs an IGKeyBagAsTable object.

Overload 1

```
public:IGKeyBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeyBagAsTable( IGKeyBagAsTable < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeyBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsTable object to another.

operator =

```
public:IGKeyBagAsTable < Element , Key , ElementOps >&
  operator =( IGKeyBagAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsTable object to another.

Miscellaneous Members

IGKeyBagAsTableTypedef

```
protected:IGKeyBagAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsTable - Type Definitions

Cursor

```
typedef IGKeyBagAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeyBagAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeyBagAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeyBagAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeyBagAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeyBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1253	os_mark_schema_type	1253

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1253		

Inherited Protected Data

None

IGKeyBagAsTableCursor

Class header file: ikbtab.h

IGKeyBagAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGKeyBagAsTableCursor

IGKeyBagAsTableMngCursor

IGKeyBagAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagAsTableCursor

Constructs an IGKeyBagAsTableCursor object.

Overload 1

```
public:IGKeyBagAsTableCursor( IGKeyBagAsTable < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeyBagAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagAsTableCursor object to another.

operator =

```
public:IGKeyBagAsTableCursor < Element , Key , ElementOps >&
  operator =( IGKeyBagAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagAsTableCursor object to another.

IGKeyBagAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeyBagCursor

Class header file: ikb.h

IGKeyBagCursor - Hierarchy List

ICursor

IElementCursor

IGKeyBagCursor

IGKeyBagMngCursor

IGKeyBagCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeyBagCursor

Constructs an IGKeyBagCursor object.

Overload 1

```
public:IGKeyBagCursor( IGKeyBag < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeyBagCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeyBagCursor object to another.

operator =

```
public:IGKeyBagCursor < Element , Key , ElementOps >&
  operator =( IGKeyBagCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeyBagCursor object to another.

IGKeyBagCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGKeySet

Variant of IKeySet with generic operations class.

Class header file: iks.h

IGKeySet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySet
IKeySet
IVGKeySet

IGKeySet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySet

Constructs an IGKeySet object.

Overload 1

```
public:IGKeySet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySet(IGKeySet < Element, Key, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySet object to another.

operator =

```
public:IGKeySet < Element , Key , ElementOps >&
  operator =( IGKeySet < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySet object to this object.

Return

A reference to this object.

IGKeySet - Type Definitions

Cursor

```
typedef IGKeySetCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Implementation

Cannot be accessed.

Variation 1

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Self

```
typedef IGKeySet < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

IGKeySet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySet < Element,
          Key, ElementOps > const& )
```

Returns a reference to the implementation object for this object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1260	os_mark_schema_type	1260

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetTypedef	1258	ImplOf	1260

Inherited Protected Data

None

IGKeySetAsAvlTree

Variant of IKeySetAsAvlTree with generic operations class.

Class header file: iksavl.h

IGKeySetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsAvlTree
IKeySetAsAvlTree
IVGKeySetAsAvlTree

IGKeySetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsAvlTree

Constructs an IGKeySetAsAvlTree object.

Overload 1

```
public:IGKeySetAsAvlTree( IGKeySetAsAvlTree < Element,
                  Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsAvlTree object to another.

operator =

```
public:IGKeySetAsAvlTree < Element , Key , ElementOps >&
  operator =( IGKeySetAsAvlTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsAvl object to this object.

Miscellaneous Members

IGKeySetAsAvlTreeTypedef

```
protected:IGKeySetAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsAvlTree - Type Definitions

Cursor

```
typedef IGKeySetAsAvlTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsAvlTreeMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsAvlTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsAvlTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1264	os_mark_schema_type	1264

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1264		

Inherited Protected Data

None

IGKeySetAsAvlTreeCursor

Class header file: iksavl.h

IGKeySetAsAvlTreeCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetAsAvlTreeCursor

IGKeySetAsAvlTreeMngCursor

IGKeySetAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsAvlTreeCursor

Overload 1

```
public:IGKeySetAsAvlTreeCursor( IGKeySetAsAvlTree < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsAvlTreeCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsAvlTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySetAsBstTree

Class header file: iksbst.h

IGKeySetAsBstTree - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAKeyCollection

IAKeySet
IGKeySetAsBstTree
IVGKeySetAsBstTree
IKeySetAsBstTree

IGKeySetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsBstTree

Constructs an IGKeySetAsBstTree object.

Overload 1

```
public:IGKeySetAsBstTree( IGKeySetAsBstTree < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsBstTree object to another.

operator =

```
public:IGKeySetAsBstTree < Element , Key , ElementOps >&
  operator =( IGKeySetAsBstTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsBstTree object to this object.

Miscellaneous Members

IGKeySetAsBstTreeTypedef

```
protected:IGKeySetAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsBstTree - Type Definitions

Cursor

```
typedef IGKeySetAsBstTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsBstTreeMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsBstTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsBstTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1269	os_mark_schema_type	1269

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1269		

Inherited Protected Data

None

IGKeySetAsBstTreeCursor

Class header file: iksbst.h

IGKeySetAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetAsBstTreeCursor

IGKeySetAsBstTreeMngCursor

IGKeySetAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsBstTreeCursor

Overload 1

```
public:IGKeySetAsBstTreeCursor( IGKeySetAsBstTree < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsBstTreeCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsBstTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySetAsDilTable

Variant of IKeySetAsDilTable with generic operations class.

Class header file: iksdil.h

IGKeySetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsDilTable
IKeySetAsDilTable
IVGKeySetAsDilTable

IGKeySetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsDilTable

Constructs an IGKeySetAsDilTable object.

Overload 1

```
public:IGKeySetAsDilTable( IGKeySetAsDilTable < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsDilTable object to another.

operator =

```
public:IGKeySetAsDilTable < Element , Key , ElementOps >&
  operator =( IGKeySetAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySetAsDilTableTypedef

```
protected:IGKeySetAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsDilTable - Type Definitions

Cursor

```
typedef IGKeySetAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsDilTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1275	os_mark_schema_type	1275

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1275		

Inherited Protected Data

None

IGKeySetAsDilTableCursor

Class header file: iksdil.h

IGKeySetAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetAsDilTableCursor

IGKeySetAsDilTableMngCursor

IGKeySetAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsDilTableCursor

Overload 1

```
public:IGKeySetAsDilTableCursor( IGKeySetAsDilTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySetAsHshTable

Variant of IKeySetAsHshTable with generic operations class.

Class header file: ikshsh.h

IGKeySetAsHshTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection

IAKeyCollection
IAKeySet
IGKeySetAsHshTable
IVGKeySetAsHshTable
IKeySetAsHshTable

IGKeySetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsHshTable

Constructs an IGKeySetAsHshTable object.

Overload 1

```
public:IGKeySetAsHshTable( IGKeySetAsHshTable < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsHshTable object to another.

operator =

```
public:IGKeySetAsHshTable < Element , Key , ElementOps >&
  operator =( IGKeySetAsHshTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsHshTable object to this object.

Miscellaneous Members

IGKeySetAsHshTableTypedef

```
protected:IGKeySetAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsHshTable - Type Definitions

Cursor

```
typedef IGKeySetAsHshTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsHshTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsHshTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsHshTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1280	os_mark_schema_type	1280

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1280		

Inherited Protected Data

None

IGKeySetAsHshTableCursor

Class header file: ikshsh.h

IGKeySetAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetAsHshTableCursor

IGKeySetAsHshTableMngCursor

IGKeySetAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsHshTableCursor

Overload 1

```
public:IGKeySetAsHshTableCursor( IGKeySetAsHshTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsHshTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsHshTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGKeySetAsList

Variant of IKeySetAsList with generic operations class.

Class header file: ikslst.h

IGKeySetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsList
IKeySetAsList
IVGKeySetAsList

IGKeySetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsList

Constructs an IGKeySetAsList object.

Overload 1

```
public:IGKeySetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySetAsList( IGKeySetAsList < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsList object to another.

operator =

```
public:IGKeySetAsList < Element , Key , ElementOps >&
  operator =( IGKeySetAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsList object to this object.

Miscellaneous Members

IGKeySetAsListTypedef

```
protected:IGKeySetAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsList - Type Definitions

Cursor

```
typedef IGKeySetAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1286	os_mark_schema_type	1286

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1286		

Inherited Protected Data

None

IGKeySetAsListCursor

Class header file: ikslst.h

IGKeySetAsListCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetAsListCursor

IGKeySetAsListMngCursor

IGKeySetAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsListCursor

Overload 1

```
public:IGKeySetAsListCursor( IGKeySetAsList < Element,
                      Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsListCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySetAsTable

Variant of IKeySetAsTable with generic operations class.

Class header file: ikstab.h

IGKeySetAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAKeyCollection

IAKeySet
IGKeySetAsTable
IVGKeySetAsTable
IKeySetAsTable

IGKeySetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySetAsTable

Constructs an IGKeySetAsTable object.

Overload 1

```
public:IGKeySetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySetAsTable( IGKeySetAsTable < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetAsTable object to another.

operator =

```
public:IGKeySetAsTable < Element , Key , ElementOps >&
  operator =( IGKeySetAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySetAsTable object to this object.

Miscellaneous Members

IGKeySetAsTableTypedef

```
protected:IGKeySetAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsTable - Type Definitions

Cursor

```
typedef IGKeySetAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySetAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySetAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySetAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySetAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1291	os_mark_schema_type	1291

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1291		

Inherited Protected Data

None

IGKeySetAsTableCursor

Class header file: ikstab.h

IGKeySetAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IGKeySetAsTableCursor
      IGKeySetAsTableMngCursor

IGKeySetAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySetAsTableCursor

Overload 1

```
public:IGKeySetAsTableCursor( IGKeySetAsTable < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySetAsTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySetAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySetAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySetCursor

Cursor class for IGKeySet

Class header file: iks.h

IGKeySetCursor - Hierarchy List

ICursor

IElementCursor

IGKeySetCursor

IGKeySetMngCursor

IGKeySetCursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IGKeySetCursor objects.

IGKeySetCursor

Constructs an IGKeySetCursor object.

Overload 1

```
public:IGKeySetCursor( IGKeySet < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySetCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySetCursor object to another.

operator =

```
public:IGKeySetCursor < Element , Key , ElementOps >&
  operator =( IGKeySetCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySetCursor object to another.

Return

A reference to this object.

IGKeySetCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGKeySortedBag

Variant of IKeySortedBag with generic operations class.

Class header file: iksb.h

IGKeySortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBag
IVGKeySortedBag
IKeySortedBag

IGKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBag

Constructs an IGKeySortedBag object.

Overload 1

```
public:IGKeySortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySortedBag( IGKeySortedBag < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBag object to another.

operator =

```
public:IGKeySortedBag < Element , Key , ElementOps >&
  operator =( IGKeySortedBag < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBag object to another.

IGKeySortedBag - Type Definitions

Cursor

```
typedef IGKeySortedBagCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Implementation

Cannot be accessed.

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedBagMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

Self

```
typedef IGKeySortedBag < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cannot be accessed.

IGKeySortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedBag < Element,
          Key, ElementOps > const& )
```

Cannot be accessed.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1298	os_mark_schema_type	1298

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagTypedef	1296	ImplOf	1298

Inherited Protected Data

None

IGKeySortedBagAsDilTable

Variant of IKeySortedBagAsDilTable with generic operations class.

Class header file: iksbdil.h

IGKeySortedBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsDilTable
IKeySortedBagAsDilTable
IVGKeySortedBagAsDilTable

IGKeySortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsDilTable

Constructs an IGKeySortedBagAsDilTable object.

Overload 1

```
public:IGKeySortedBagAsDilTable( IGKeySortedBagAsDilTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsDilTable object to another.

operator =

```
public:IGKeySortedBagAsDilTable < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsDilTable object to another.

Miscellaneous Members

IGKeySortedBagAsDilTableTypedef

```
protected:IGKeySortedBagAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsDilTable - Type Definitions

Cursor

```
typedef IGKeySortedBagAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedBagAsDilTableMngCursor < Element , Key , ElementOps >
  MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedBagAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedBagAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag IAKeySortedBag	921 922	compare	923

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1302	os_mark_schema_type	1303

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1303		

Inherited Protected Data

None

IGKeySortedBagAsDilTableCursor

Class header file: iksbdil.h

IGKeySortedBagAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGKeySortedBagAsDilTableCursor
        IGKeySortedBagAsDilTableMngCursor

IGKeySortedBagAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsDilTableCursor

Constructs an IGKeySortedBagAsDilTableCursor object.

Overload 1

```
public:IGKeySortedBagAsDilTableCursor( IGKeySortedBagAsDilTable < Element,
                                       Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedBagAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsDilTableCursor object to another.

operator =

```
public:IGKeySortedBagAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsDilTableCursor object to another.

IGKeySortedBagAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedBagAsList

Variant of IKeySortedBagAsList with generic operations class.

Class header file: iksblst.h

IGKeySortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsList
IKeySortedBagAsList
IVGKeySortedBagAsList

IGKeySortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsList

Constructs an IGKeySortedBagAsList object.

Overload 1

```
public:IGKeySortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySortedBagAsList( IGKeySortedBagAsList < Element,
                        Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsList object to another.

operator =

```
public:IGKeySortedBagAsList < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsList object to another.

Miscellaneous Members

IGKeySortedBagAsListTypedef

```
protected:IGKeySortedBagAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsList - Type Definitions

Cursor

```
typedef IGKeySortedBagAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedBagAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedBagAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedBagAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1309	os_mark_schema_type	1309

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1309		

Inherited Protected Data

None

IGKeySortedBagAsListCursor

Class header file: iksblst.h

IGKeySortedBagAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGKeySortedBagAsListCursor
        IGKeySortedBagAsListMngCursor

IGKeySortedBagAsListCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsListCursor

Constructs an IGKeySortedBagAsListCursor object.

Overload 1

```
public:IGKeySortedBagAsListCursor( IGKeySortedBagAsList < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedBagAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsListCursor object to another.

operator =

```
public:IGKeySortedBagAsListCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsListCursor object to another.

IGKeySortedBagAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGKeySortedBagAsTable

Variant of IKeySortedBagAsTable with generic operations class.

Class header file: iksbtab.h

IGKeySortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsTable
IVGKeySortedBagAsTable
IKeySortedBagAsTable

IGKeySortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsTable

Constructs an IGKeySortedBagAsTable object.

Overload 1

```
public:IGKeySortedBagAsTable( IGKeySortedBagAsTable < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsTable object to another.

operator =

```
public:IGKeySortedBagAsTable < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsTable object to another.

Miscellaneous Members

IGKeySortedBagAsTableTypedef

```
protected:IGKeySortedBagAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsTable - Type Definitions

Cursor

```
typedef IGKeySortedBagAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedBagAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedBagAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedBagAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1315	os_mark_schema_type	1315

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1315		

Inherited Protected Data

None

IGKeySortedBagAsTableCursor

Class header file: iksbtab.h

IGKeySortedBagAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGKeySortedBagAsTableCursor
        IGKeySortedBagAsTableMngCursor

IGKeySortedBagAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagAsTableCursor

Constructs an IGKeySortedBagAsTableCursor object.

Overload 1

```
public:IGKeySortedBagAsTableCursor( IGKeySortedBagAsTable < Element,
                                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedBagAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagAsTableCursor object to another.

operator =

```
public:IGKeySortedBagAsTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedBagAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagAsTableCursor object to another.

IGKeySortedBagAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedBagCursor

Cursor class for IGKeySortedBag.

Class header file: iksb.h

IGKeySortedBagCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGKeySortedBagCursor
        IGKeySortedBagMngCursor

IGKeySortedBagCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedBagCursor

Constructs an IGKeySortedBagCursor object.

Overload 1

```
public:IGKeySortedBagCursor( IGKeySortedBag < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedBagCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedBagCursor object to another.

operator =

```
public:IGKeySortedBagCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedBagCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGKeySortedBagCursor object to another.

IGKeySortedBagCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGKeySortedSet

Variant of IKeySortedSet with generic operations class.

Class header file: ikss.h

IGKeySortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSet
IKeySortedSet
IVGKeySortedSet

IGKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSet

Constructs an IGKeySortedSet object.

Overload 1

```
public:IGKeySortedSet( IGKeySortedSet < Element,
                       Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSet object to another.

operator =

```
public:IGKeySortedSet < Element , Key , ElementOps >&
  operator =( IGKeySortedSet < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSet object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetTypedef

```
protected:IGKeySortedSetTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSet - Type Definitions

Cursor

```
typedef IGKeySortedSetCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSet < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSet < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1323	os_mark_schema_type	1323

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1323		

Inherited Protected Data

None

IGKeySortedSetAsAvlTree

Variant of IKeySortedSetAsAvlTree with generic operations class.

Class header file: ikssavl.h

IGKeySortedSetAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeySortedCollection
        - IAKeySortedSet
          - **IGKeySortedSetAsAvlTree**
            - IVGKeySortedSetAsAvlTree
            - IKeySortedSetAsAvlTree

IGKeySortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSetAsAvlTree

Constructs an IGKeySortedSetAsAvlTree object.

Overload 1

```
public:IGKeySortedSetAsAvlTree( IGKeySortedSetAsAvlTree < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSetAsAvlTree object to another.

operator =

```
public:IGKeySortedSetAsAvlTree < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsAvlTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSetAsAvlTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetAsAvlTreeTypedef

```
protected:IGKeySortedSetAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsAvlTree - Type Definitions

Cursor

```
typedef IGKeySortedSetAsAvlTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetAsAvlTreeMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSetAsAvlTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSetAsAvlTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1327	os_mark_schema_type	1327

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1327		

Inherited Protected Data

None

IGKeySortedSetAsAvlTreeCursor

Class header file: ikssavl.h

IGKeySortedSetAsAvlTreeCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetAsAvlTreeCursor

IGKeySortedSetAsAvlTreeMngCursor

IGKeySortedSetAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetAsAvlTreeCursor

Overload 1

```
public:IGKeySortedSetAsAvlTreeCursor( IGKeySortedSetAsAvlTree < Element,
                                      Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetAsAvlTreeCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsAvlTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedSetAsBstTree

Variant of IKeySortedSetAsBstTree with generic operations class.

Class header file: ikssbst.h

IGKeySortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsBstTree
IKeySortedSetAsBstTree
IVGKeySortedSetAsBstTree

IGKeySortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSetAsBstTree

Constructs an IGKeySortedSetAsBstTree object.

Overload 1

```
public:IGKeySortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySortedSetAsBstTree( IGKeySortedSetAsBstTree < Element,
                          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSetAsBstTree object to another.

operator =

```
public:IGKeySortedSetAsBstTree < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsBstTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSetAsBstTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetAsBstTreeTypedef

```
protected:IGKeySortedSetAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsBstTree - Type Definitions

Cursor

```
typedef IGKeySortedSetAsBstTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetAsBstTreeMngCursor < Element , Key ,
ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSetAsBstTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSetAsBstTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1333	os_mark_schema_type	1333

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1333		

Inherited Protected Data

None

IGKeySortedSetAsBstTreeCursor

Class header file: ikssbst.h

IGKeySortedSetAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetAsBstTreeCursor

IGKeySortedSetAsBstTreeMngCursor

IGKeySortedSetAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetAsBstTreeCursor

Overload 1

```
public:IGKeySortedSetAsBstTreeCursor( IGKeySortedSetAsBstTree < Element,
                                      Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetAsBstTreeCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsBstTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedSetAsDilTable

Variant of IKeySortedSetAsDilTable with generic operations class.

Class header file: ikssdil.h

IGKeySortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsDilTable
IKeySortedSetAsDilTable
IVGKeySortedSetAsDilTable

IGKeySortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSetAsDilTable

Constructs an IGKeySortedSetAsDilTable object.

Overload 1

```
public:IGKeySortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGKeySortedSetAsDilTable( IGKeySortedSetAsDilTable < Element,
                                 Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGKeySortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSetAsDilTable object to another.

operator =

```
public:IGKeySortedSetAsDilTable < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSetAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetAsDilTableTypedef

```
protected:IGKeySortedSetAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsDilTable - Type Definitions

Cursor

```
typedef IGKeySortedSetAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetAsDilTableMngCursor < Element , Key ,
ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSetAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSetAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1339	os_mark_schema_type	1339

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1339		

Inherited Protected Data

None

IGKeySortedSetAsDilTableCursor

Class header file: ikssdil.h

IGKeySortedSetAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetAsDilTableCursor

IGKeySortedSetAsDilTableMngCursor

IGKeySortedSetAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetAsDilTableCursor

Overload 1

```
public:IGKeySortedSetAsDilTableCursor( IGKeySortedSetAsDilTable
                                       < Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedSetAsList

Variant of IKeySortedSetAsList with generic operations class.

Class header file: iksslst.h

IGKeySortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsList
IVGKeySortedSetAsList
IKeySortedSetAsList

IGKeySortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSetAsList

Constructs an IGKeySortedSetAsList object.

Overload 1

```
public:IGKeySortedSetAsList( IGKeySortedSetAsList < Element,
                        Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSetAsList object to another.

operator =

```
public:IGKeySortedSetAsList < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSetAsList object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetAsListTypedef

```
protected:IGKeySortedSetAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsList - Type Definitions

Cursor

```
typedef IGKeySortedSetAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetAsListMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSetAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSetAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1345	os_mark_schema_type	1345

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1345		

Inherited Protected Data

None

IGKeySortedSetAsListCursor

Class header file: iksslst.h

IGKeySortedSetAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetAsListCursor

IGKeySortedSetAsListMngCursor

IGKeySortedSetAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetAsListCursor

Overload 1

```
public:IGKeySortedSetAsListCursor( IGKeySortedSetAsList < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetAsListCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedSetAsTable

Variant of IKeySortedSetAsTable with generic operations class.

Class header file: iksstab.h

IGKeySortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsTable
IVGKeySortedSetAsTable
IKeySortedSetAsTable

IGKeySortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGKeySortedSetAsTable

Constructs an IGKeySortedSetAsTable object.

Overload 1

```
public:IGKeySortedSetAsTable( IGKeySortedSetAsTable < Element,
                        Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGKeySortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGKeySortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGKeySortedSetAsTable object to another.

operator =

```
public:IGKeySortedSetAsTable < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGKeySortedSetAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGKeySortedSetAsTableTypedef

```
protected:IGKeySortedSetAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsTable - Type Definitions

Cursor

```
typedef IGKeySortedSetAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGKeySortedSetAsTableMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedSetAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGKeySortedSetAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGKeySortedSetAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1351	os_mark_schema_type	1351

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1351		

Inherited Protected Data

None

IGKeySortedSetAsTableCursor

Class header file: iksstab.h

IGKeySortedSetAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetAsTableCursor

IGKeySortedSetAsTableMngCursor

IGKeySortedSetAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetAsTableCursor

Overload 1

```
public:IGKeySortedSetAsTableCursor( IGKeySortedSetAsTable < Element,
                                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetAsTableCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGKeySortedSetCursor

Class header file: ikss.h

IGKeySortedSetCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGKeySortedSetCursor

IGKeySortedSetMngCursor

IGKeySortedSetCursor - Member Functions and Data by Group

Constructors & Destructor

IGKeySortedSetCursor

Overload 1

```
public:IGKeySortedSetCursor( IGKeySortedSet < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGKeySortedSetCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGKeySortedSetCursor < Element , Key , ElementOps >&
  operator =( IGKeySortedSetCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGKeySortedSetCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGMap

Variant of IMap with generic operations class.

Class header file: imap.h

IGMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMap
IVGMap
IMap

IGMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMap

Constructs an IGMap object.

Overload 1

```
public:IGMap(IGMap < Element, Key, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMap object to another.

operator =

```
public:IGMap < Element , Key , ElementOps >&
  operator =( IGMap < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMap object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapTypedef

```
protected:IGMapTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMap - Type Definitions

Cursor

```
typedef IGMapCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMap < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMap < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1359	os_mark_schema_type	1359

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1359		

Inherited Protected Data

None

IGMapAsAvlTree

Variant of IMapAsAvlTree with generic operations class.

Class header file: imapavl.h

IGMapAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IAMap
          - **IGMapAsAvlTree**
            - IMapAsAvlTree
            - IVGMapAsAvlTree

IGMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsAvlTree

Constructs an IGMapAsAvlTree object.

Overload 1

```
public:IGMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGMapAsAvlTree( IGMapAsAvlTree < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMapAsAvlTree object to another.

operator =

```
public:IGMapAsAvlTree < Element , Key , ElementOps >&
  operator =( IGMapAsAvlTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsAvlTreeTypedef

```
protected:IGMapAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsAvlTree - Type Definitions

Cursor

```
typedef IGMapAsAvlTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsAvlTreeMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsAvlTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsAvlTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1363	os_mark_schema_type	1363

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1363		

Inherited Protected Data

None

IGMapAsAvlTreeCursor

Class header file: imapavl.h

IGMapAsAvlTreeCursor - Hierarchy List

ICursor

IElementCursor

IGMapAsAvlTreeCursor

IGMapAsAvlTreeMngCursor

IGMapAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsAvlTreeCursor

Overload 1

```
public:IGMapAsAvlTreeCursor( IGMapAsAvlTree < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsAvlTreeCursor < Element , Key , ElementOps >&
  operator =( IGMapAsAvlTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGMapAsBstTree

Variant of IMapAsBstTree with generic operations class.

Class header file: imapbst.h

IGMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsBstTree
IVGMapAsBstTree
IMapAsBstTree

IGMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsBstTree

Constructs an IGMapAsBstTree object.

Overload 1

```
public:IGMapAsBstTree( IGMapAsBstTree < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMapAsBstTree object to another.

operator =

```
public:IGMapAsBstTree < Element , Key , ElementOps >&
  operator =( IGMapAsBstTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMapAsBstTree object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsBstTreeTypedef

```
protected:IGMapAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsBstTree - Type Definitions

Cursor

```
typedef IGMapAsBstTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsBstTreeMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsBstTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsBstTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1369	os_mark_schema_type	1369

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1369		

Inherited Protected Data

None

IGMapAsBstTreeCursor

Class header file: imapbst.h

IGMapAsBstTreeCursor - Hierarchy List

ICursor
  IElementCursor
    IGMapAsBstTreeCursor
      IGMapAsBstTreeMngCursor

IGMapAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsBstTreeCursor

Overload 1

```
public:IGMapAsBstTreeCursor( IGMapAsBstTree < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsBstTreeCursor < Element , Key , ElementOps >&
  operator =( IGMapAsBstTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGMapAsDilTable

Variant of IMapAsDilTable with generic operations class.

Class header file: imapdil.h

IGMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsDilTable
IMapAsDilTable
IVGMapAsDilTable

IGMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsDilTable

Constructs an IGMapAsDilTable object.

Overload 1

```
public:IGMapAsDilTable( IGMapAsDilTable < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMapAsDilTable object to another.

operator =

```
public:IGMapAsDilTable < Element , Key , ElementOps >&
  operator =( IGMapAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMapAsDilTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsDilTableTypedef

```
protected:IGMapAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsDilTable - Type Definitions

Cursor

```
typedef IGMapAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsDilTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1374	os_mark_schema_type	1375

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1374		

Inherited Protected Data

None

IGMapAsDilTableCursor

Class header file: imapdil.h

IGMapAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGMapAsDilTableCursor

IGMapAsDilTableMngCursor

IGMapAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsDilTableCursor

Overload 1

```
public:IGMapAsDilTableCursor( IGMapAsDilTable < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGMapAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGMapAsHshTable

Variant of IMapAsHshTable with generic operations class.

Class header file: imaphsh.h

IGMapAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsHshTable
IMapAsHshTable
IVGMapAsHshTable

IGMapAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsHshTable

Constructs an IGMapAsHshTable object.

Overload 1

```
public:IGMapAsHshTable( IGMapAsHshTable < Element,
                        Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGMapAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGMapAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMapAsHshTable object to another.

operator =

```
public:IGMapAsHshTable < Element , Key , ElementOps >&
  operator =( IGMapAsHshTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMapAsHshTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsHshTableTypedef

```
protected:IGMapAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsHshTable - Type Definitions

Cursor

```
typedef IGMapAsHshTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsHshTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsHshTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsHshTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	938	unionWith	938

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1380	os_mark_schema_type	1380

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	941		

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1380		

Inherited Protected Data

None

IGMapAsHshTableCursor

Class header file: imaphsh.h

IGMapAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGMapAsHshTableCursor

IGMapAsHshTableMngCursor

IGMapAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsHshTableCursor

Overload 1

```
public:IGMapAsHshTableCursor( IGMapAsHshTable < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsHshTableCursor < Element , Key , ElementOps >&
  operator =( IGMapAsHshTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGMapAsList

Variant of IMapAsList with generic operations class.

Class header file: imaplst.h

IGMapAsList - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection

IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsList
IMapAsList
IVGMapAsList

IGMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsList

Constructs an IGMapAsList object.

Overload 1

```
public:IGMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGMapAsList( IGMapAsList < Element,
            Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMap object to another.

operator =

```
public:IGMapAsList < Element , Key , ElementOps >&
  operator =( IGMapAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMapAsList object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsListTypedef

```
protected:IGMapAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsList - Type Definitions

Cursor

```
typedef IGMapAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	938	unionWith	938

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1386	os_mark_schema_type	1386

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	941		

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1386		

Inherited Protected Data

None

IGMapAsListCursor

Class header file: imaplst.h

IGMapAsListCursor - Hierarchy List

ICursor

IElementCursor

IGMapAsListCursor

IGMapAsListMngCursor

IGMapAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsListCursor

Overload 1

```
public:IGMapAsListCursor( IGMapAsList < Element,
                          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsListCursor < Element , Key , ElementOps >&
  operator =( IGMapAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGMapAsTable

Variant of IMapAsTable with generic operations class.

Class header file: imaptab.h

IGMapAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection

IAEqualityKeyCollection
IAMap
IGMapAsTable
IVGMapAsTable
IMapAsTable

IGMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGMapAsTable

Constructs an IGMapAsTable object.

Overload 1

```
public:IGMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGMapAsTable( IGMapAsTable < Element,
                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGMapAsTable object to another.

operator =

```
public:IGMapAsTable < Element , Key , ElementOps >&
  operator =( IGMapAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGMapAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGMapAsTableTypedef

```
protected:IGMapAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsTable - Type Definitions

Cursor

```
typedef IGMapAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMapAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMapAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGMapAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGMapAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1392	os_mark_schema_type	1392

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1392		

Inherited Protected Data

None

IGMapAsTableCursor

Class header file: imaptab.h

IGMapAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGMapAsTableCursor

IGMapAsTableMngCursor

IGMapAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapAsTableCursor

Overload 1

```
public:IGMapAsTableCursor( IGMapAsTable < Element,
                           Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapAsTableCursor < Element , Key , ElementOps >&
  operator =( IGMapAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGMapCursor

Class header file: imap.h

IGMapCursor - Hierarchy List

ICursor

IElementCursor

IGMapCursor

IGMapMngCursor

IGMapCursor - Member Functions and Data by Group

Constructors & Destructor

IGMapCursor

Overload 1

```
public:IGMapCursor(IGMap < Element, Key, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMapCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMapCursor < Element , Key , ElementOps >&
  operator =( IGMapCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMapCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGMultiwayTree

IGMultiwayTree is the default implementation variant with generic operations class.

Class header file: imwt.h

IGMultiwayTree - Hierarchy List

IMetaTree
IATreeBase
IATree
IAMultiwayTree
IGMultiwayTree
IVGMultiwayTree
IMultiwayTree
IGTree

IGMultiwayTree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IGMultiwayTree

Constructs an IGMultiwayTree object.

Overload 1

```
public:IGMultiwayTree( IGMultiwayTree < numOfChildren,
                       Element,
                       ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGMultiwayTree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGMultiwayTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMultiwayTree - Type Definitions

Cursor

```
typedef IGMultiwayTreeCursor < numOfChildren , Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Guard

```
typedef ITreeCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Implementation

Do not access.

Variation 1

```
typedef ICMultiwayTreeImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGMultiwayTreeMngCursor < numOfChildren , Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IMultiwayTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Self

```
typedef IGMultiwayTree < numOfChildren , Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

IGMultiwayTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGMultiwayTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946
hasChild	947	numberOfChildren	947

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1399	operator =	1398
os_mark_schema_type	1399		

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTreeTypedef	1398	ImplOf	1398

Inherited Protected Data

None

IGMultiwayTreeCursor

Cursor class for IGMultiwayTree collections.

Class header file: imwt.h

IGMultiwayTreeCursor - Hierarchy List

ITreeCursor

IElementTreeCursor

IGMultiwayTreeCursor

IGMultiwayTreeMngCursor

IGMultiwayTreeCursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IGMultiwayTreeCursor

Constructs an IGMultiwayTreeCursor object.

Overload 1

```
public:IGMultiwayTreeCursor( IGMultiwayTree < numOfChildren,
                             Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGMultiwayTreeCursor(ITreeCursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGMultiwayTreeCursor < numOfChildren , Element , ElementOps >&
  operator =( IGMultiwayTreeCursor < numOfChildren,
              Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not use this operator.

IGMultiwayTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ITreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ITreeCursor	2039	copy	2042
invalidate	2042	isValid	2042
operator !=	2043	operator =	2043
operator ==	2043	setToChild	2040
setToFirstExistingChild	2040	setToLastExistingChild	2040
setToNextExistingChild	2041	setToParent	2041
setToPreviousExistingChild	2041	setToRoot	2042

IElementTreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1117		

Inherited Public Data

None

Inherited Protected Functions

IElementTreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementTreeCursor	1117		

ITreeCursor			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	2043	ITreeCursor	2039

Inherited Protected Data

None

IGPriorityQueue

Variant of IPriorityQueue with generic operations class.

Class header file: ipqu.h

IGPriorityQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueue
IPriorityQueue
IVGPriorityQueue

IGPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGPriorityQueue

Constructs an IGPriorityQueue object.

Overload 1

```
public:IGPriorityQueue( IGPriorityQueue < Element,
                        Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGPriorityQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGPriorityQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGPriorityQueue object to another.

operator =

```
public:IGPriorityQueue < Element , Key , ElementOps >&
  operator =( IGPriorityQueue < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGPriorityQueue object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGPriorityQueueTypedef

```
protected:IGPriorityQueueTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueue - Type Definitions

Cursor

```
typedef IGPriorityQueueCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGPriorityQueueMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGPriorityQueue < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGPriorityQueue < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1404	os_mark_schema_type	1404

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1404		

Inherited Protected Data

None

IGPriorityQueueAsDilTable

Class header file: ipqudil.h

IGPriorityQueueAsDilTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAPriorityQueue
        IGPriorityQueueAsDilTable
          IPriorityQueueAsDilTable

IGPriorityQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGPriorityQueueAsDilTable

Constructs an IGPriorityQueueAsDilTable object.

Overload 1

```
public:IGPriorityQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGPriorityQueueAsDilTable( IGPriorityQueueAsDilTable < Element,
                                  Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGPriorityQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGPriorityQueueAsDilTable object to another.

operator =

```
public:IGPriorityQueueAsDilTable < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGPriorityQueueAsDilTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGPriorityQueueAsDilTableTypedef

```
protected:IGPriorityQueueAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsDilTable - Type Definitions

Cursor

```
typedef IGPriorityQueueAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGPriorityQueueAsDilTableMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGPriorityQueueAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGPriorityQueueAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1408	os_mark_schema_type	1408

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1408		

Inherited Protected Data

None

IGPriorityQueueAsDilTableCursor

Class header file: ipqudil.h

IGPriorityQueueAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGPriorityQueueAsDilTableCursor
        IGPriorityQueueAsDilTableMngCursor

IGPriorityQueueAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGPriorityQueueAsDilTableCursor

Overload 1

```
public:IGPriorityQueueAsDilTableCursor( IGPriorityQueueAsDilTable
                                        < Element , Key,
                                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGPriorityQueueAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGPriorityQueueAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGPriorityQueueAsList

Variant of IPriorityQueueAsList with generic operations class.

Class header file: ipqulst.h

IGPriorityQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsList
IVGPriorityQueueAsList
IPriorityQueueAsList

IGPriorityQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGPriorityQueueAsList

Constructs an IGPriorityQueueAsList object.

Overload 1

```
public:IGPriorityQueueAsList( IGPriorityQueueAsList < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGPriorityQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGPriorityQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGPriorityQueueAsList object to another.

operator =

```
public:IGPriorityQueueAsList < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGPriorityQueueAsList object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGPriorityQueueAsListTypedef

```
protected:IGPriorityQueueAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsList - Type Definitions

Cursor

```
typedef IGPriorityQueueAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGPriorityQueueAsListMngCursor < Element ,
                                         Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGPriorityQueueAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGPriorityQueueAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1414	os_mark_schema_type	1414

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1414		

Inherited Protected Data

None

IGPriorityQueueAsListCursor

Class header file: ipqulst.h

IGPriorityQueueAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGPriorityQueueAsListCursor

IGPriorityQueueAsListMngCursor

IGPriorityQueueAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGPriorityQueueAsListCursor

Overload 1

```
public:IGPriorityQueueAsListCursor( IGPriorityQueueAsList < Element,
                                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGPriorityQueueAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGPriorityQueueAsListCursor < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGPriorityQueueAsTable

Variant of IPriorityQueueAsTable with generic operations class.

Class header file: ipqutab.h

IGPriorityQueueAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsTable
IPriorityQueueAsTable
IVGPriorityQueueAsTable

IGPriorityQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGPriorityQueueAsTable

Constructs an IGPriorityQueueAsTable object.

Overload 1

```
public:IGPriorityQueueAsTable( IGPriorityQueueAsTable < Element,
                               Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGPriorityQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGPriorityQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGPriorityQueueAsTable object to another.
operator =

```
public:IGPriorityQueueAsTable < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGPriorityQueueAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGPriorityQueueAsTableTypedef

```
protected:IGPriorityQueueAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsTable - Type Definitions

Cursor

```
typedef IGPriorityQueueAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGPriorityQueueAsTableMngCursor < Element , Key ,
ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IKeySortedBagAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGPriorityQueueAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGPriorityQueueAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1419	os_mark_schema_type	1419

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1419		

Inherited Protected Data

None

IGPriorityQueueAsTableCursor

Class header file: ipqutab.h

IGPriorityQueueAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGPriorityQueueAsTableCursor

IGPriorityQueueAsTableMngCursor

IGPriorityQueueAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGPriorityQueueAsTableCursor

Overload 1

```
public:IGPriorityQueueAsTableCursor( IGPriorityQueueAsTable < Element,
                                     Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGPriorityQueueAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGPriorityQueueAsTableCursor < Element , Key , ElementOps >&
  operator =( IGPriorityQueueAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGPriorityQueueCursor

Class header file: ipqu.h

IGPriorityQueueCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGPriorityQueueCursor

IGPriorityQueueMngCursor

IGPriorityQueueCursor - Member Functions and Data by Group

Constructors & Destructor

IGPriorityQueueCursor

Overload 1

```
public:IGPriorityQueueCursor( IGPriorityQueue < Element,
                              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGPriorityQueueCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGPriorityQueueCursor < Element , Key , ElementOps >&
  operator =( IGPriorityQueueCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGPriorityQueueCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGQueue

Variant of IQueue with generic operations class.

Class header file: iqu.h

IGQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueue
IQueue
IVGQueue

IGQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGQueue

Constructs an IGQueue object.

Overload 1

```
public:IGQueue(IGQueue < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGQueue object to another.

operator =

```
public:IGQueue < Element , ElementOps >&
  operator =( IGQueue < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGQueue object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGQueueTypedef

```
protected:IGQueueTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueue - Type Definitions

Cursor

```
typedef IGQueueCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGQueueMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGQueue < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGQueue < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueue - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToPrevious	985		

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1426	os_mark_schema_type	1427

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1426		

Inherited Protected Data

None

IGQueueAsDilTable

Variant of IQueueAsDilTable with generic operations class.

Class header file: iqudil.h

IGQueueAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsDilTable
IQueueAsDilTable
IVGQueueAsDilTable

IGQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGQueueAsDilTable

Constructs an IGQueueAsDilTable object.

Overload 1

```
public:IGQueueAsDilTable( IGQueueAsDilTable < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGQueueAsDilTable object to another.

operator =

```
public:IGQueueAsDilTable < Element , ElementOps >&
  operator =( IGQueueAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGQueueAsDilTable object to another.

Miscellaneous Members

IGQueueAsDilTableTypedef

```
protected:IGQueueAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsDilTable - Type Definitions

Cursor

```
typedef IGQueueAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGQueueAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGQueueAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGQueueAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1430	os_mark_schema_type	1430

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1430		

Inherited Protected Data

None

IGQueueAsDilTableCursor

Class header file: iqudil.h

IGQueueAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGQueueAsDilTableCursor
        IGQueueAsDilTableMngCursor

IGQueueAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGQueueAsDilTableCursor

Overload 1

```
public:IGQueueAsDilTableCursor( IGQueueAsDilTable < Element,
                                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGQueueAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGQueueAsDilTableCursor < Element , ElementOps >&
  operator =( IGQueueAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGQueueAsList

Variant of IQueueAsList with generic operations class.

Class header file: iqulst.h

IGQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsList
IQueueAsList
IVGQueueAsList

IGQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGQueueAsList

Constructs an IGQueueAsList object.

Overload 1

```
public:IGQueueAsList(IGQueueAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsListTypedef

```
protected:IGQueueAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGQueueAsList object to another.

operator =

```
public:IGQueueAsList < Element , ElementOps >&
  operator =( IGQueueAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGQueueAsList object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGQueueAsListTypedef

```
protected:IGQueueAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsList - Type Definitions

Cursor

```
typedef IGQueueAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGQueueAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGQueueAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGQueueAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1436	os_mark_schema_type	1436

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1436		

Inherited Protected Data

None

IGQueueAsListCursor

Class header file: iqulst.h

IGQueueAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGQueueAsListCursor

IGQueueAsListMngCursor

IGQueueAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGQueueAsListCursor

Overload 1

```
public:IGQueueAsListCursor( IGQueueAsList < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGQueueAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGQueueAsListCursor < Element , ElementOps >&
  operator =( IGQueueAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGQueueAsTable

Variant of IQueueAsTable with generic operations class.

Class header file: iqutab.h

IGQueueAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAQueue
        IGQueueAsTable
          IQueueAsTable

IGQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGQueueAsTable

Constructs an IGQueueAsTable object.

Overload 1

```
public:IGQueueAsTable( IGQueueAsTable < Element,
                       ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGQueueAsTable object to another.

operator =

```
public:IGQueueAsTable < Element , ElementOps >&
  operator =( IGQueueAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGQueueAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGQueueAsTableTypedef

```
protected:IGQueueAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsTable - Type Definitions

Cursor

```
typedef IGQueueAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGQueueAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGQueueAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGQueueAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1441	os_mark_schema_type	1441

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1441		

Inherited Protected Data

None

IGQueueAsTableCursor

Class header file: iqutab.h

IGQueueAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGQueueAsTableCursor
        IGQueueAsTableMngCursor

IGQueueAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGQueueAsTableCursor

Overload 1

```
public:IGQueueAsTableCursor( IGQueueAsTable < Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGQueueAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGQueueAsTableCursor < Element , ElementOps >&
  operator =( IGQueueAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGQueueCursor

Class header file: iqu.h

IGQueueCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGQueueCursor
        IGQueueMngCursor

IGQueueCursor - Member Functions and Data by Group

Constructors & Destructor

IGQueueCursor

Overload 1

```
public:IGQueueCursor(IGQueue < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGQueueCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGQueueCursor < Element , ElementOps >&
  operator =( IGQueueCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGQueueCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGRelation

Variant of IRelation with generic operations class.

Class header file: irel.h

IGRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelation
IRelation
IVGRelation

IGRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGRelation

Constructs an IGRelation object.

Overload 1

```
public:IGRelation(IGRelation < Element, Key, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGRelation object to another.
operator =

```
public:IGRelation < Element , Key , ElementOps >&
  operator =( IGRelation < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGRelation object to another.

Miscellaneous Members

IGRelationTypedef

```
protected:IGRelationTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelation - Type Definitions

Cursor

```
typedef IGRelationCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGRelationMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IRelationAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGRelation < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGRelation < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	973	unionWith	975

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1448	os_mark_schema_type	1448

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1448		

Inherited Protected Data

None

IGRelationAsDilTable

Class header file: ireldil.h

IGRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsDilTable
IRelationAsDilTable
IVGRelationAsDilTable

IGRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGRelationAsDilTable

Constructs an IGRelationAsDilTable object.

Overload 1

```
public:IGRelationAsDilTable( IGRelationAsDilTable < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGRelationAsDilTable object to another.

operator =

```
public:IGRelationAsDilTable < Element , Key , ElementOps >&
  operator =( IGRelationAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGRelationAsDilTable object to another.

Miscellaneous Members

IGRelationAsDilTableTypedef

```
protected:IGRelationAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsDilTable - Type Definitions

Cursor

```
typedef IGRelationAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGRelationAsDilTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IRelationAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGRelationAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGRelationAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1452	os_mark_schema_type	1453

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1452		

Inherited Protected Data

None

IGRelationAsDilTableCursor

Class header file: ireldil.h

IGRelationAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGRelationAsDilTableCursor

IGRelationAsDilTableMngCursor

IGRelationAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGRelationAsDilTableCursor

Overload 1

```
public:IGRelationAsDilTableCursor( IGRelationAsDilTable < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGRelationAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGRelationAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGRelationAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGRelationAsHshTable

Class header file: irelhsh.h

IGRelationAsHshTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection

IAEqualityKeyCollection
IARelation
IGRelationAsHshTable
IVGRelationAsHshTable
IRelationAsHshTable

IGRelationAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGRelationAsHshTable

Constructs an IGRelationAsHshTable object.

Overload 1

```
public:IGRelationAsHshTable( IGRelationAsHshTable < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGRelationAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGRelationAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGRelationAsHshTable object to another.

operator =

```
public:IGRelationAsHshTable < Element , Key , ElementOps >&
  operator =( IGRelationAsHshTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGRelationAsHshTable object to another.

Miscellaneous Members

IGRelationAsHshTableTypedef

```
protected:IGRelationAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsHshTable - Type Definitions

Cursor

```
typedef IGRelationAsHshTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGRelationAsHshTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IRelationAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGRelationAsHshTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGRelationAsHshTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1458	os_mark_schema_type	1458

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1458		

Inherited Protected Data

None

IGRelationAsHshTableCursor

Class header file: irelhsh.h

IGRelationAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGRelationAsHshTableCursor

IGRelationAsHshTableMngCursor

IGRelationAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGRelationAsHshTableCursor

Overload 1

```
public:IGRelationAsHshTableCursor( IGRelationAsHshTable < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGRelationAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGRelationAsHshTableCursor < Element , Key , ElementOps >&
  operator =( IGRelationAsHshTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGRelationAsList

Class header file: irellst.h

IGRelationAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsList
IRelationAsList
IVGRelationAsList

IGRelationAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGRelationAsList

Constructs an IGRelationAsList object.

Overload 1

```
public:IGRelationAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGRelationAsList( IGRelationAsList < Element,
                         Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGRelationAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGRelationAsList object to another.

operator =

```
public:IGRelationAsList < Element , Key , ElementOps >&
  operator =( IGRelationAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGRelationAsList object to another.

Miscellaneous Members

IGRelationAsListTypedef

```
protected:IGRelationAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsList - Type Definitions

Cursor

```
typedef IGRelationAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGRelationAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IRelationAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGRelationAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGRelationAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1464	os_mark_schema_type	1464

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1464		

Inherited Protected Data

None

IGRelationAsListCursor

Class header file: irellst.h

IGRelationAsListCursor - Hierarchy List

ICursor

IElementCursor

IGRelationAsListCursor

IGRelationAsListMngCursor

IGRelationAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGRelationAsListCursor

Overload 1

```
public:IGRelationAsListCursor( IGRelationAsList < Element,
                               Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGRelationAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGRelationAsListCursor < Element , Key , ElementOps >&
  operator =( IGRelationAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGRelationAsTable

Class header file: ireltab.h

IGRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsTable
IRelationAsTable
IVGRelationAsTable

IGRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGRelationAsTable

Constructs an IGRelationAsTable object.

Overload 1

```
public:IGRelationAsTable( IGRelationAsTable < Element,
                          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGRelationAsTable object to another.

operator =

```
public:IGRelationAsTable < Element , Key , ElementOps >&
  operator =( IGRelationAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGRelationAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGRelationAsTableTypedef

```
protected:IGRelationAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsTable - Type Definitions

Cursor

```
typedef IGRelationAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGRelationAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef IRelationAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGRelationAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGRelationAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1469	os_mark_schema_type	1469

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1469		

Inherited Protected Data

None

IGRelationAsTableCursor

Class header file: ireltab.h

IGRelationAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGRelationAsTableCursor

IGRelationAsTableMngCursor

IGRelationAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGRelationAsTableCursor

Overload 1

```
public:IGRelationAsTableCursor( IGRelationAsTable < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGRelationAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGRelationAsTableCursor < Element , Key , ElementOps >&
  operator =( IGRelationAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGRelationCursor

Class header file: irel.h

IGRelationCursor - Hierarchy List

ICursor

IElementCursor

IGRelationCursor

IGRelationMngCursor

IGRelationCursor - Member Functions and Data by Group

Constructors & Destructor

IGRelationCursor

Overload 1

```
public:IGRelationCursor( IGRelation < Element,
                         Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGRelationCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGRelationCursor < Element , Key , ElementOps >&
  operator =( IGRelationCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGRelationCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGSequence

Variant of ISequence with generic operations class.

Class header file: iseq.h

IGSequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequence
ISequence
IVGSequence

IGSequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSequence

Constructs an IGSequence object.

Overload 1

```
public:IGSequence(IGSequence < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSequence object to another.

operator =

```
public:IGSequence < Element , ElementOps >&
  operator =( IGSequence < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSequence object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSequenceTypedef

```
protected:IGSequenceTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequence - Type Definitions

Cursor

```
typedef IGSequenceCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSequenceMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSequence < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSequence < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1476	os_mark_schema_type	1477

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
`IASequence`	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IASequentialCollection`	996		

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1476		

Inherited Protected Data

None

IGSequenceAsDilTable

Variant of ISequenceAsDilTable with generic operations class.

Class header file: iseqdil.h

IGSequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsDilTable
IVGSequenceAsDilTable
ISequenceAsDilTable

IGSequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSequenceAsDilTable

Constructs an IGSequenceAsDilTable object.

Overload 1

```
public:IGSequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSequenceAsDilTable( IGSequenceAsDilTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSequenceAsDilTable object to another.

operator =

```
public:IGSequenceAsDilTable < Element , ElementOps >&
  operator =( IGSequenceAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSequenceAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSequenceAsDilTableTypedef

```
protected:IGSequenceAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsDilTable - Type Definitions

Cursor

```
typedef IGSequenceAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSequenceAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSequenceAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSequenceAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1481	os_mark_schema_type	1481

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1481		

Inherited Protected Data

None

IGSequenceAsDilTableCursor

Class header file: iseqdil.h

IGSequenceAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSequenceAsDilTableCursor
        IGSequenceAsDilTableMngCursor

IGSequenceAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSequenceAsDilTableCursor

Overload 1

```
public:IGSequenceAsDilTableCursor( IGSequenceAsDilTable < Element,
                                   ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSequenceAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSequenceAsDilTableCursor < Element , ElementOps >&
  operator =( IGSequenceAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSequenceAsList

Variant of ISequenceAsList with generic operations class.

Class header file: iseqlst.h

IGSequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsList
IVGSequenceAsList
ISequenceAsList

IGSequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSequenceAsList

Constructs an IGSequenceAsList object.

Overload 1

```
public:IGSequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSequenceAsList( IGSequenceAsList < Element,
                  ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSequenceAsList object to another.

operator =

```
public:IGSequenceAsList < Element , ElementOps >&
  operator =( IGSequenceAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSequenceAsList object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSequenceAsListTypedef

```
protected:IGSequenceAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsList - Type Definitions

Cursor

```
typedef IGSequenceAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSequenceAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSequenceAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSequenceAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1487	os_mark_schema_type	1487

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1487		

Inherited Protected Data

None

IGSequenceAsListCursor

Class header file: iseqlst.h

IGSequenceAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSequenceAsListCursor
        IGSequenceAsListMngCursor

IGSequenceAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGSequenceAsListCursor

Overload 1

```
public:IGSequenceAsListCursor( IGSequenceAsList < Element,
                               ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSequenceAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSequenceAsListCursor < Element , ElementOps >&
  operator =( IGSequenceAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSequenceAsTable

Variant of ISequenceAsTable with generic operations class.

Class header file: iseqtab.h

IGSequenceAsTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAOrderedCollection
      - IASequentialCollection
        - IASequence
          - **IGSequenceAsTable**
            - ISequenceAsTable
            - IVGSequenceAsTable

IGSequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSequenceAsTable

Constructs an IGSequenceAsTable object.

Overload 1

```
public:IGSequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSequenceAsTable( IGSequenceAsTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSequenceAsTable object to another.

operator =

```
public:IGSequenceAsTable < Element , ElementOps >&
  operator =( IGSequenceAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSequenceAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSequenceAsTableTypedef

```
protected:IGSequenceAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsTable - Type Definitions

Cursor

```
typedef IGSequenceAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSequenceAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSequenceAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSequenceAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1493	os_mark_schema_type	1493

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1493		

Inherited Protected Data

None

IGSequenceAsTableCursor

Class header file: iseqtab.h

IGSequenceAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSequenceAsTableCursor
        IGSequenceAsTableMngCursor

IGSequenceAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSequenceAsTableCursor

Overload 1

```
public:IGSequenceAsTableCursor( IGSequenceAsTable < Element,
                                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSequenceAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSequenceAsTableCursor < Element , ElementOps >&
  operator =( IGSequenceAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSequenceCursor

Class header file: iseq.h

IGSequenceCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSequenceCursor
        IGSequenceMngCursor

IGSequenceCursor - Member Functions and Data by Group

Constructors & Destructor

IGSequenceCursor

Overload 1

```
public:IGSequenceCursor(IGSequence < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSequenceCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSequenceCursor < Element , ElementOps >&
  operator =( IGSequenceCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSequenceCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSet

Variant of ISet with generic operations class.

Class header file: iset.h

IGSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSet
ISet
IVGSet

IGSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSet

Constructs an IGSet object.

Overload 1

```
public:IGSet(IGSet < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSet object to another.

operator =

```
public:IGSet < Element , ElementOps >&
  operator =( IGSet < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSet object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetTypedef

```
protected:IGSetTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSet - Type Definitions

Cursor

```
typedef IGSetCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSet < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSet < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1500	os_mark_schema_type	1501

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1500		

Inherited Protected Data

None

IGSetAsAvlTree

Class header file: isetavl.h

IGSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsAvlTree
IVGSetAsAvlTree
ISetAsAvlTree

IGSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsAvlTree

Constructs an IGSetAsAvlTree object.

Overload 1

```
public:IGSetAsAvlTree( IGSetAsAvlTree < Element,
                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsAvlTree object to another.

operator =

```
public:IGSetAsAvlTree < Element , ElementOps >&
  operator =( IGSetAsAvlTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsAvlTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsAvlTreeTypedef

```
protected:IGSetAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsAvlTree - Type Definitions

Cursor

```
typedef IGSetAsAvlTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsAvlTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsAvlTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsAvlTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1504	os_mark_schema_type	1505

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1504		

Inherited Protected Data

None

IGSetAsAvlTreeCursor

Class header file: isetavl.h

IGSetAsAvlTreeCursor - Hierarchy List

ICursor

IElementCursor

IGSetAsAvlTreeCursor

IGSetAsAvlTreeMngCursor

IGSetAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsAvlTreeCursor

Overload 1

```
public:IGSetAsAvlTreeCursor( IGSetAsAvlTree < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsAvlTreeCursor < Element , ElementOps >&
  operator =( IGSetAsAvlTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**

`operator !=`
`operator ==`
`setToNext`
	1101
	1101
	1101
		`operator =`
		`setToFirst`
			1100
			1101

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`element`	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGSetAsBstTree

Variant of ISetAsBstTree with generic operations class.

Class header file: isetbst.h

IGSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsBstTree
ISetAsBstTree
IVGSetAsBstTree

IGSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsBstTree

Constructs an IGSetAsBstTree object.

Overload 1

```
public:IGSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSetAsBstTree( IGSetAsBstTree < Element,
                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsBstTree object to another.

operator =

```
public:IGSetAsBstTree < Element , ElementOps >&
  operator =( IGSetAsBstTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsBstTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsBstTreeTypedef

```
protected:IGSetAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsBstTree - Type Definitions

Cursor

```
typedef IGSetAsBstTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsBstTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsBstTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsBstTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1510	os_mark_schema_type	1510

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
`IASet`	1003	`locateNext`	1008
`numberOfOccurrences`	1009	`removeAllOccurrences`	1009

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1510		

Inherited Protected Data

None

IGSetAsBstTreeCursor

Class header file: isetbst.h

IGSetAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IGSetAsBstTreeCursor

IGSetAsBstTreeMngCursor

IGSetAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsBstTreeCursor

Overload 1

```
public:IGSetAsBstTreeCursor( IGSetAsBstTree < Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsBstTreeCursor < Element , ElementOps >&
  operator =( IGSetAsBstTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGSetAsDilTable

Variant of ISetAsDilTable with generic operations class.

Class header file: isetdil.h

IGSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsDilTable
IVGSetAsDilTable
ISetAsDilTable

IGSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsDilTable

Constructs an IGSetAsDilTable object.

Overload 1

```
public:IGSetAsDilTable( IGSetAsDilTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsDilTable object to another.

operator =

```
public:IGSetAsDilTable < Element , ElementOps >&
  operator =( IGSetAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsDilTableTypedef

```
protected:IGSetAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsDilTable - Type Definitions

Cursor

```
typedef IGSetAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1515	os_mark_schema_type	1515

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1515		

Inherited Protected Data

None

IGSetAsDilTableCursor

Class header file: isetdil.h

IGSetAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IGSetAsDilTableCursor

IGSetAsDilTableMngCursor

IGSetAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsDilTableCursor

Overload 1

```
public:IGSetAsDilTableCursor( IGSetAsDilTable < Element,
                              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsDilTableCursor < Element , ElementOps >&
  operator =( IGSetAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGSetAsHshTable

Variant of ISetAsHshTable with generic operations class.

Class header file: isethsh.h

IGSetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsHshTable
ISetAsHshTable
IVGSetAsHshTable

IGSetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsHshTable

Constructs an IGSetAsHshTable object.

Overload 1

```
public:IGSetAsHshTable( IGSetAsHshTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsHshTable object to another.

operator =

```
public:IGSetAsHshTable < Element , ElementOps >&
  operator =( IGSetAsHshTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsHshTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsHshTableTypedef

```
protected:IGSetAsHshTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsHshTable - Type Definitions

Cursor

```
typedef IGSetAsHshTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsHshTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsHshTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsHshTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsHshTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1521	os_mark_schema_type	1521

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet numberOfOccurrences	1003 1009	locateNext removeAllOccurrences	1008 1009

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1521		

Inherited Protected Data

None

IGSetAsHshTableCursor

Class header file: isethsh.h

IGSetAsHshTableCursor - Hierarchy List

ICursor

IElementCursor

IGSetAsHshTableCursor

IGSetAsHshTableMngCursor

IGSetAsHshTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsHshTableCursor

Overload 1

```
public:IGSetAsHshTableCursor( IGSetAsHshTable < Element,
                              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsHshTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsHshTableCursor < Element , ElementOps >&
  operator =( IGSetAsHshTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsHshTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGSetAsList

Variant of ISetAsList with generic operations class.

Class header file: isetlst.h

IGSetAsList - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection

IAEqualityCollection
IASet
IGSetAsList
ISetAsList
IVGSetAsList

IGSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsList

Constructs an IGSetAsList object.

Overload 1

```
public:IGSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSetAsList(IGSetAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsList object to another.

operator =

```
public:IGSetAsList < Element , ElementOps >&
  operator =( IGSetAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsList object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsListTypedef

```
protected:IGSetAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsList - Type Definitions

Cursor

```
typedef IGSetAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1526	os_mark_schema_type	1526

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1526		

Inherited Protected Data

None

IGSetAsListCursor

Class header file: isetlst.h

IGSetAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IGSetAsListCursor
      IGSetAsListMngCursor

IGSetAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsListCursor

Overload 1

```
public:IGSetAsListCursor( IGSetAsList < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsListCursor < Element , ElementOps >&
  operator =( IGSetAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

Inherited Protected Data

None

IGSetAsTable

Variant of ISetAsTable with generic operations class.

Class header file: isettab.h

IGSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsTable
ISetAsTable
IVGSetAsTable

IGSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSetAsTable

Constructs an IGSetAsTable object.

Overload 1

```
public:IGSetAsTable(IGSetAsTable < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSetAsTable object to another.

operator =

```
public:IGSetAsTable < Element , ElementOps >&
  operator =( IGSetAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSetAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSetAsTableTypedef

```
protected:IGSetAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsTable - Type Definitions

Cursor

```
typedef IGSetAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSetAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISetAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSetAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSetAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1531	os_mark_schema_type	1532

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1531		

Inherited Protected Data

None

IGSetAsTableCursor

Class header file: isettab.h

IGSetAsTableCursor - Hierarchy List

ICursor

IElementCursor

IGSetAsTableCursor

IGSetAsTableMngCursor

IGSetAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetAsTableCursor

Overload 1

```
public:IGSetAsTableCursor( IGSetAsTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetAsTableCursor < Element , ElementOps >&
  operator =( IGSetAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGSetCursor

Class header file: iset.h

IGSetCursor - Hierarchy List

ICursor

IElementCursor

IGSetCursor

IGSetMngCursor

IGSetCursor - Member Functions and Data by Group

Constructors & Destructor

IGSetCursor

Overload 1

```
public:IGSetCursor(IGSet < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSetCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSetCursor < Element , ElementOps >&
  operator =( IGSetCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSetCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IGSortedBag

Variant of ISortedBag with generic operations class.

Class header file: isb.h

IGSortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBag
IVGSortedBag
ISortedBag

IGSortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBag

Constructs an IGSortedBag object.

Overload 1

```
public:IGSortedBag(IGSortedBag < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBag object to another.

operator =

```
public:IGSortedBag < Element , ElementOps >&
  operator =( IGSortedBag < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBag object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedBagTypedef

```
protected:IGSortedBagTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBag - Type Definitions

Cursor

```
typedef IGSortedBagCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBag < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBag < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1538	os_mark_schema_type	1538

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1538		

Inherited Protected Data

None

IGSortedBagAsAvlTree

Variant of ISortedBagAsAvlTree with generic operations class.

Class header file: isbavl.h

IGSortedBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsAvlTree
ISortedBagAsAvlTree
IVGSortedBagAsAvlTree

IGSortedBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsAvlTree

Constructs an IGSortedBagAsAvlTree object.

Overload 1

```
public:IGSortedBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedBagAsAvlTree( IGSortedBagAsAvlTree < Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsAvlTree object to another.

operator =

```
public:IGSortedBagAsAvlTree < Element , ElementOps >&
  operator =( IGSortedBagAsAvlTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsAvlTree object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedBagAsAvlTreeTypedef

```
protected:IGSortedBagAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsAvlTree - Type Definitions

Cursor

```
typedef IGSortedBagAsAvlTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagAsAvlTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBagAsAvlTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBagAsAvlTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1543	os_mark_schema_type	1543

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualitySortedCollection`	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
`IASortedBag`	1011		

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1543		

Inherited Protected Data

None

IGSortedBagAsAvlTreeCursor

Class header file: isbavl.h

IGSortedBagAsAvlTreeCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedBagAsAvlTreeCursor
        IGSortedBagAsAvlTreeMngCursor

IGSortedBagAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsAvlTreeCursor

Constructs an IGSortedBagAsAvlTreeCursor object.

Overload 1

```
public:IGSortedBagAsAvlTreeCursor( IGSortedBagAsAvlTree < Element,
                                   ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsAvlTreeCursor object to another.

operator =

```
public:IGSortedBagAsAvlTreeCursor < Element , ElementOps >&
  operator =( IGSortedBagAsAvlTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsAvlTreeCursor object to another.

IGSortedBagAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedBagAsBstTree

Variant of ISortedBagAsBstTree with generic operations class.

Class header file: isbbst.h

IGSortedBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsBstTree
IVGSortedBagAsBstTree
ISortedBagAsBstTree

IGSortedBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsBstTree

Constructs an IGSortedBagAsBstTree object.

Overload 1

```
public:IGSortedBagAsBstTree( IGSortedBagAsBstTree < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsBstlTree object to another.

operator =

```
public:IGSortedBagAsBstTree < Element , ElementOps >&
  operator =( IGSortedBagAsBstTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsBstTree object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedBagAsBstTreeTypedef

```
protected:IGSortedBagAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsBstTree - Type Definitions

Cursor

```
typedef IGSortedBagAsBstTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagAsBstTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBagAsBstTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBagAsBstTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1549	os_mark_schema_type	1549

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1549		

Inherited Protected Data

None

IGSortedBagAsBstTreeCursor

Class header file: isbbst.h

IGSortedBagAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedBagAsBstTreeCursor

IGSortedBagAsBstTreeMngCursor

IGSortedBagAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsBstTreeCursor

Constructs an IGSortedBagAsBstTreeCursor object.

Overload 1

```
public:IGSortedBagAsBstTreeCursor( IGSortedBagAsBstTree < Element,
                                   ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsBstTreeCursor object to another.

operator =

```
public:IGSortedBagAsBstTreeCursor < Element , ElementOps >&
  operator =( IGSortedBagAsBstTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsBstTreeCursor object to another.

IGSortedBagAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedBagAsDilTable

Variant of ISortedBagAsDilTable with generic operations class.

Class header file: isbdil.h

IGSortedBagAsDilTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection
        IAEqualitySortedCollection
          IASortedBag

IGSortedBagAsDilTable

ISortedBagAsDilTable

IVGSortedBagAsDilTable

IGSortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsDilTable

Constructs an IGSortedBagAsDilTable object.

Overload 1

```
public:IGSortedBagAsDilTable( IGSortedBagAsDilTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsDilTable object to another.

operator =

```
public:IGSortedBagAsDilTable < Element , ElementOps >&
  operator =( IGSortedBagAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsDilTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedBagAsDilTableTypedef

```
protected:IGSortedBagAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsDilTable - Type Definitions

Cursor

```
typedef IGSortedBagAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBagAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBagAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1555	os_mark_schema_type	1555

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1555		

Inherited Protected Data

None

IGSortedBagAsDilTableCursor

Class header file: isbdil.h

IGSortedBagAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedBagAsDilTableCursor

IGSortedBagAsDilTableMngCursor

IGSortedBagAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsDilTableCursor

Constructs an IGSortedBagAsDilTableCursor object.

Overload 1

```
public:IGSortedBagAsDilTableCursor( IGSortedBagAsDilTable < Element,
                                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsDilTableCursor object to another.

operator =

```
public:IGSortedBagAsDilTableCursor < Element , ElementOps >&
  operator =( IGSortedBagAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsDilTableCursor object to another.

IGSortedBagAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedBagAsList

Variant of ISortedBagAsList with generic operations class.

Class header file: isblst.h

IGSortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsList
IVGSortedBagAsList
ISortedBagAsList

IGSortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGSortedBagAsList

Constructs an IGSortedBagAsList object.

Overload 1

```
public:IGSortedBagAsList( IGSortedBagAsList < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGBagAsAvlTree object to another.
operator =

```
public:IGSortedBagAsList < Element , ElementOps >&
  operator =( IGSortedBagAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsList object to another.

Protected Type Definitions

Protected type definitions associated with this class.
IGSortedBagAsListTypedef

```
protected:IGSortedBagAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsList - Type Definitions

Cursor

```
typedef IGSortedBagAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBagAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBagAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag numberOfDifferentElements operator == unionWith	1011 1016 1016 1017	intersectionWith operator != setToNextDifferentElement	1015 1016 1016

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1561	os_mark_schema_type	1561

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1561		

Inherited Protected Data

None

IGSortedBagAsListCursor

Class header file: isblst.h

IGSortedBagAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedBagAsListCursor
        IGSortedBagAsListMngCursor

IGSortedBagAsListCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsListCursor

Constructs an IGSortedBagAsListCursor object.

Overload 1

```
public:IGSortedBagAsListCursor( IGSortedBagAsList < Element,
                                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsListCursor object to another.

operator =

```
public:IGSortedBagAsListCursor < Element , ElementOps >&
  operator =( IGSortedBagAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsListCursor object to another.

IGSortedBagAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedBagAsTable

Variant of ISortedBagAsTable with generic operations class.

Class header file: isbtab.h

IGSortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsTable
ISortedBagAsTable
IVGSortedBagAsTable

IGSortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGSortedBagAsTable

Constructs an IGSortedBagAsTable object.

Overload 1

```
public:IGSortedBagAsTable( IGSortedBagAsTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsTable object to another.
operator =

```
public:IGSortedBagAsTable < Element , ElementOps >&
  operator =( IGSortedBagAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedBagAsTableTypedef

```
protected:IGSortedBagAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsTable - Type Definitions

Cursor

```
typedef IGSortedBagAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedBagAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedBagAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedBagAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedBagAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1567	os_mark_schema_type	1567

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1567		

Inherited Protected Data

None

IGSortedBagAsTableCursor

Class header file: isbtab.h

IGSortedBagAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedBagAsTableCursor

IGSortedBagAsTableMngCursor

IGSortedBagAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagAsTableCursor

Constructs an IGSortedBagAsTableCursor object.

Overload 1

```
public:IGSortedBagAsTableCursor( IGSortedBagAsTable < Element,
                                 ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagAsTableCursor object to another.

operator =

```
public:IGSortedBagAsTableCursor < Element , ElementOps >&
  operator =( IGSortedBagAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagAsTableCursor object to another.

IGSortedBagAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedBagCursor

Class header file: isb.h

IGSortedBagCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedBagCursor

IGSortedBagMngCursor

IGSortedBagCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedBagCursor

Constructs an IGSortedBagCursor object.

Overload 1

```
public:IGSortedBagCursor( IGSortedBag < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedBagCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedBagCursor object to another.

operator =

```
public:IGSortedBagCursor < Element , ElementOps >&
  operator =( IGSortedBagCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGSortedBagCursor object to another.

IGSortedBagCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedMap

Variant of ISortedMap with generic operations class.

Class header file: ism.h

IGSortedMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMap
ISortedMap
IVGSortedMap

IGSortedMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedMap

Constructs an IGSortedMap object.

Overload 1

```
public:IGSortedMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedMap( IGSortedMap < Element,
                    Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMap object to another.

operator =

```
public:IGSortedMap < Element , Key , ElementOps >&
  operator =( IGSortedMap < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMap object to this object.

Return

A reference to this object.

IGSortedMap - Type Definitions

Cursor

```
typedef IGSortedMapCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Implementation

Do not access.

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Self

```
typedef IGSortedMap < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

IGSortedMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMap < Element,
          Key, ElementOps > const& )
```

Return a reference to the implementation object for this object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1575	os_mark_schema_type	1576

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapTypedef	1574	ImplOf	1575

Inherited Protected Data

None

IGSortedMapAsAvlTree

Variant of ISortedMapAsAvlTree with generic operations class.

Class header file: ismavl.h

IGSortedMapAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsAvlTree
ISortedMapAsAvlTree
IVGSortedMapAsAvlTree

IGSortedMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedMapAsAvlTree

Constructs an IGSortedMapAsAvlTree object.

Overload 1

```
public:IGSortedMapAsAvlTree( IGSortedMapAsAvlTree < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMapAsAvlTree object to another.

operator =

```
public:IGSortedMapAsAvlTree < Element , Key , ElementOps >&
  operator =( IGSortedMapAsAvlTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMapAsAvlTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedMapAsAvlTreeTypedef

```
protected:IGSortedMapAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsAvlTree - Type Definitions

Cursor

```
typedef IGSortedMapAsAvlTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapAsAvlTreeMngCursor < Element , Key ,
                                        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedMapAsAvlTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMapAsAvlTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1580	os_mark_schema_type	1580

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1580		

Inherited Protected Data

None

IGSortedMapAsAvlTreeCursor

Class header file: ismavl.h

IGSortedMapAsAvlTreeCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedMapAsAvlTreeCursor
        IGSortedMapAsAvlTreeMngCursor

IGSortedMapAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedMapAsAvlTreeCursor

Overload 1

```
public:IGSortedMapAsAvlTreeCursor( IGSortedMapAsAvlTree < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedMapAsAvlTreeCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapAsAvlTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedMapAsBstTree

Variant of ISortedMapAsBstTree with generic operations class.

Class header file: ismbst.h

IGSortedMapAsBstTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - **IGSortedMapAsBstTree**
            - ISortedMapAsBstTree
            - IVGSortedMapAsBstTree

IGSortedMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedMapAsBstTree

Constructs an IGSortedMapAsBstTree object.

Overload 1

```
public:IGSortedMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedMapAsBstTree( IGSortedMapAsBstTree < Element,
                             Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMapAsBstTree object to another.

operator =

```
public:IGSortedMapAsBstTree < Element , Key , ElementOps >&
  operator =( IGSortedMapAsBstTree < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMapAsBstTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedMapAsBstTreeTypedef

```
protected:IGSortedMapAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsBstTree - Type Definitions

Cursor

```
typedef IGSortedMapAsBstTreeCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapAsBstTreeMngCursor < Element , Key ,
                                        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedMapAsBstTree < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMapAsBstTree < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1586	os_mark_schema_type	1586

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1586		

Inherited Protected Data

None

IGSortedMapAsBstTreeCursor

Class header file: ismbst.h

IGSortedMapAsBstTreeCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedMapAsBstTreeCursor

IGSortedMapAsBstTreeMngCursor

IGSortedMapAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedMapAsBstTreeCursor

Overload 1

```
public:IGSortedMapAsBstTreeCursor( IGSortedMapAsBstTree <
                                   Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedMapAsBstTreeCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapAsBstTreeCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedMapAsDilTable

Variant of ISortedMapAsDilTable with generic operations class.

Class header file: ismdil.h

IGSortedMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsDilTable
IVGSortedMapAsDilTable
ISortedMapAsDilTable

IGSortedMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedMapAsDilTable

Constructs an IGSortedMapAsDilTable object.

Overload 1

```
public:IGSortedMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedMapAsDilTable( IGSortedMapAsDilTable < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMapAsDilTable object to another.

operator =

```
public:IGSortedMapAsDilTable < Element , Key , ElementOps >&
  operator =( IGSortedMapAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMapAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedMapAsDilTableTypedef

```
protected:IGSortedMapAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsDilTable - Type Definitions

Cursor

```
typedef IGSortedMapAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapAsDilTableMngCursor < Element , Key ,
                                         ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedMapAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMapAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1592	os_mark_schema_type	1592

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1592		

Inherited Protected Data

None

IGSortedMapAsDilTableCursor

Class header file: ismdil.h

IGSortedMapAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedMapAsDilTableCursor

IGSortedMapAsDilTableMngCursor

IGSortedMapAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedMapAsDilTableCursor

Overload 1

```
public:IGSortedMapAsDilTableCursor( IGSortedMapAsDilTable <
                                    Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedMapAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`operator ==` `setToNext`	1101 1101	`setToFirst`	1101

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`element`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`setToLast`	1810	`setToPrevious`	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedMapAsList

Variant of ISortedMapAsList with generic operations class.

Class header file: ismlst.h

IGSortedMapAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsList
ISortedMapAsList
IVGSortedMapAsList

IGSortedMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGSortedMapAsList

Constructs an IGSortedMapAsList object.

Overload 1

```
public:IGSortedMapAsList( IGSortedMapAsList < Element,
                          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMapAsList object to another.
operator =

```
public:IGSortedMapAsList < Element , Key , ElementOps >&
  operator =( IGSortedMapAsList < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMapAsList object to this object.

Protected Type Definitions

Protected type definitions associated with this class.
IGSortedMapAsListTypedef

```
protected:IGSortedMapAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsList - Type Definitions

Cursor

```
typedef IGSortedMapAsListCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapAsListMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedMapAsList < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMapAsList < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1598	os_mark_schema_type	1598

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap numberOfElementsWithKey	1023 1028	numberOfDifferentKeys numberOfOccurrences	1028 1028

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
removeAllElementsWithKey setToNextWithDifferentKey	1029 1029	removeAllOccurrences	1029

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1598		

Inherited Protected Data

None

IGSortedMapAsListCursor

Class header file: ismlst.h

IGSortedMapAsListCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedMapAsListCursor
        IGSortedMapAsListMngCursor

IGSortedMapAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedMapAsListCursor

Overload 1

```
public:IGSortedMapAsListCursor( IGSortedMapAsList
                                  < Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedMapAsListCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapAsListCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedMapAsTable

Variant of ISortedMapAsTable with generic operations class.

Class header file: ismtab.h

IGSortedMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsTable
IVGSortedMapAsTable
ISortedMapAsTable

IGSortedMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedMapAsTable

Constructs an IGSortedMapAsTable object.

Overload 1

```
public:IGSortedMapAsTable( IGSortedMapAsTable < Element,
                           Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedMapAsTable object to another.

operator =

```
public:IGSortedMapAsTable < Element , Key , ElementOps >&
  operator =( IGSortedMapAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedMapAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedMapAsTableTypedef

```
protected:IGSortedMapAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsTable - Type Definitions

Cursor

```
typedef IGSortedMapAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedMapAsTableMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedMapAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedMapAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedMapAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1604	os_mark_schema_type	1604

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	1023 1028 1029 1029	numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	1028 1028 1029

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1604		

Inherited Protected Data

None

IGSortedMapAsTableCursor

Class header file: ismtab.h

IGSortedMapAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedMapAsTableCursor
        IGSortedMapAsTableMngCursor

IGSortedMapAsTableCursor - Member Functions and Data by Group

Constructors & Destructor
IGSortedMapAsTableCursor

Overload 1

```
public:IGSortedMapAsTableCursor( IGSortedMapAsTable
                                 < Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator
operator =

```
public:IGSortedMapAsTableCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedMapAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedMapCursor

Cursor for a IGSortedMap collection.

Class header file: ism.h

IGSortedMapCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedMapCursor
        IGSortedMapMngCursor

IGSortedMapCursor - Member Functions and Data by Group

Constructors & Destructor

Construct and destruct IGSortedMapCursor objects.

IGSortedMapCursor

Construct an IGSortedMapCursor object.

Overload 1

```
public:IGSortedMapCursor( IGSortedMap < Element,
                          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedMapCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

Assign another sorted map cursor to this object.

operator =

```
public:IGSortedMapCursor < Element , Key , ElementOps >&
  operator =( IGSortedMapCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign another cursor object to this object.

IGSortedMapCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedRelation

Variant of ISortedRelation with generic operations class.

Class header file: isr.h

IGSortedRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelation
ISortedRelation
IVGSortedRelation

IGSortedRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IGSortedRelation

Constructs an IGSortedRelation object.

Overload 1

```
public:IGSortedRelation( IGSortedRelation < Element,
                         Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedRelation object to another.
operator =

```
public:IGSortedRelation < Element , Key , ElementOps >&
  operator =( IGSortedRelation < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedRelation object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedRelationTypedef

```
protected:IGSortedRelationTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelation - Type Definitions

Cursor

```
typedef IGSortedRelationCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedRelationMngCursor < Element , Key , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedRelationAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedRelation < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedRelation < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1612	os_mark_schema_type	1612

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1612		

Inherited Protected Data

None

IGSortedRelationAsDilTable

Variant of ISortedRelationAsDilTable with generic operations class.

Class header file: isrdil.h

IGSortedRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsDilTable
ISortedRelationAsDilTable
IVGSortedRelationAsDilTable

IGSortedRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedRelationAsDilTable

Constructs an IGSortedRelationAsDilTable object.

Overload 1

```
public:IGSortedRelationAsDilTable( IGSortedRelationAsDilTable < Element,
                                   Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedRelationAsDilTable object to another.

operator =

```
public:IGSortedRelationAsDilTable < Element , Key , ElementOps >&
  operator =( IGSortedRelationAsDilTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedRelationAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedRelationAsDilTableTypedef

```
protected:IGSortedRelationAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsDilTable - Type Definitions

Cursor

```
typedef IGSortedRelationAsDilTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedRelationAsDilTableMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedRelationAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedRelationAsDilTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedRelationAsDilTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1617	os_mark_schema_type	1617

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1617		

Inherited Protected Data

None

IGSortedRelationAsDilTableCursor

Class header file: isrdil.h

IGSortedRelationAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedRelationAsDilTableCursor
        IGSortedRelationAsDilTableMngCursor

IGSortedRelationAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor
IGSortedRelationAsDilTableCursor

Overload 1

```
public:IGSortedRelationAsDilTableCursor( IGSortedRelationAsDilTable
                                         < Element , Key,
                                         ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedRelationAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedRelationAsDilTableCursor < Element , Key , ElementOps >&
  operator =( IGSortedRelationAsDilTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedRelationAsTable

Variant of ISortedRelationAsTable with generic operations class.

Class header file: isrtab.h

IGSortedRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsTable
ISortedRelationAsTable
IVGSortedRelationAsTable

IGSortedRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedRelationAsTable

Constructs an IGSortedRelationAsTable object.

Overload 1

```
public:IGSortedRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedRelationAsTable( IGSortedRelationAsTable < Element,
                                Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedRelationAsTable object to another.

operator =

```
public:IGSortedRelationAsTable < Element , Key , ElementOps >&
  operator =( IGSortedRelationAsTable < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedRelationAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedRelationAsTableTypedef

```
protected:IGSortedRelationAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsTable - Type Definitions

Cursor

```
typedef IGSortedRelationAsTableCursor < Element , Key , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedRelationAsTableMngCursor < Element , Key ,
        ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedRelationAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedRelationAsTable < Element , Key , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedRelationAsTable < Element,
          Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1623	os_mark_schema_type	1623

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1623		

Inherited Protected Data

None

IGSortedRelationAsTableCursor

Class header file: isrtab.h

IGSortedRelationAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedRelationAsTableCursor
        IGSortedRelationAsTableMngCursor

IGSortedRelationAsTableCursor - Member Functions and Data by Group

Constructors & Destructor
IGSortedRelationAsTableCursor

Overload 1

```
public:IGSortedRelationAsTableCursor( IGSortedRelationAsTable
                                      < Element, Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedRelationAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedRelationAsTableCursor < Element , Key , ElementOps >&
  operator =( IGSortedRelationAsTableCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedRelationCursor

Class header file: isr.h

IGSortedRelationCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedRelationCursor

IGSortedRelationMngCursor

IGSortedRelationCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedRelationCursor

Overload 1

```
public:IGSortedRelationCursor( IGSortedRelation < Element,
                               Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedRelationCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedRelationCursor < Element , Key , ElementOps >&
  operator =( IGSortedRelationCursor < Element,
              Key, ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedRelationCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedSet

Variant of ISortedSet with generic operations class.

Class header file: iss.h

IGSortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSet
ISortedSet
IVGSortedSet

IGSortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSet

Constructs an IGSortedSet object.

Overload 1

```
public:IGSortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedSet(IGSortedSet < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetTypedef

```
protected:IGSortedSetTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an IGSortedSetTypedef object.

Assignment Operator

This operator assigns one IGSortedSet object to another.

operator =

```
public:IGSortedSet < Element , ElementOps >&
  operator =( IGSortedSet < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSet object to this object.

Return

A reference to this object.

Implementation

These members provide access to object implementation details.

ImplOf

```
protected:static Implementation&
  ImplOf( IGSortedSet < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Return a reference to the implementation object for this object.

IGSortedSet - Type Definitions

Cursor

```
typedef IGSortedSetCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Implementation

Do not access.

Variation 1

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

Self

```
typedef IGSortedSet < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Do not access.

IGSortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1631	os_mark_schema_type	1631

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

Inherited Protected Data

None

IGSortedSetAsAvlTree

Variant of ISortedSetAsAvlTree with generic operations class.

Class header file: issavl.h

IGSortedSetAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedSet
          - **IGSortedSetAsAvlTree**
            - IVGSortedSetAsAvlTree
            - ISortedSetAsAvlTree

IGSortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetAsAvlTree

Constructs an IGSortedSetAsAvlTree object.

Overload 1

```
public:IGSortedSetAsAvlTree( IGSortedSetAsAvlTree < Element,
                             ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetAsAvlTree object to another.

operator =

```
public:IGSortedSetAsAvlTree < Element , ElementOps >&
  operator =( IGSortedSetAsAvlTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSetAsAvlTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedSetAsAvlTreeTypedef

```
protected:IGSortedSetAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsAvlTree - Type Definitions

Cursor

```
typedef IGSortedSetAsAvlTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetAsAvlTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsAvlTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedSetAsAvlTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedSetAsAvlTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1635	os_mark_schema_type	1635

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1635		

Inherited Protected Data

None

IGSortedSetAsAvlTreeCursor

Class header file: issavl.h

IGSortedSetAsAvlTreeCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedSetAsAvlTreeCursor
        IGSortedSetAsAvlTreeMngCursor

IGSortedSetAsAvlTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedSetAsAvlTreeCursor

Overload 1

```
public:IGSortedSetAsAvlTreeCursor( IGSortedSetAsAvlTree < Element,
                                   ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetAsAvlTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedSetAsAvlTreeCursor < Element , ElementOps >&
  operator =( IGSortedSetAsAvlTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsAvlTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedSetAsBstTree

Class header file: issbst.h

IGSortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsBstTree
ISortedSetAsBstTree
IVGSortedSetAsBstTree

IGSortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetAsBstTree

Constructs an IGSortedSetAsBstTree object.

Overload 1

```
public:IGSortedSetAsBstTree( IGSortedSetAsBstTree < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetAsBstTree object to another.

operator =

```
public:IGSortedSetAsBstTree < Element , ElementOps >&
  operator =( IGSortedSetAsBstTree < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSetAsBstTree object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedSetAsBstTreeTypedef

```
protected:IGSortedSetAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsBstTree - Type Definitions

Cursor

```
typedef IGSortedSetAsBstTreeCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetAsBstTreeMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsBstTreeImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedSetAsBstTree < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedSetAsBstTree < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1641	os_mark_schema_type	1641

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1641		

Inherited Protected Data

None

IGSortedSetAsBstTreeCursor

Class header file: issbst.h

IGSortedSetAsBstTreeCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGSortedSetAsBstTreeCursor
        IGSortedSetAsBstTreeMngCursor

IGSortedSetAsBstTreeCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedSetAsBstTreeCursor

Overload 1

```
public:IGSortedSetAsBstTreeCursor( IGSortedSetAsBstTree < Element,
                                   ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetAsBstTreeCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedSetAsBstTreeCursor < Element , ElementOps >&
  operator =( IGSortedSetAsBstTreeCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsBstTreeCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGSortedSetAsDilTable

Variant of ISortedSetAsDilTable with generic operations class.

Class header file: issdil.h

IGSortedSetAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedSet
          - **IGSortedSetAsDilTable**
            - IVGSortedSetAsDilTable
            - ISortedSetAsDilTable

IGSortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetAsDilTable

Constructs an IGSortedSetAsDilTable object.

Overload 1

```
public:IGSortedSetAsDilTable( IGSortedSetAsDilTable < Element,
                              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetAsDilTable object to another.

operator =

```
public:IGSortedSetAsDilTable < Element , ElementOps >&
  operator =( IGSortedSetAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSetAsDilTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedSetAsDilTableTypedef

```
protected:IGSortedSetAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsDilTable - Type Definitions

Cursor

```
typedef IGSortedSetAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedSetAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedSetAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1647	os_mark_schema_type	1647

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1647		

Inherited Protected Data

None

IGSortedSetAsDilTableCursor

Class header file: issdil.h

IGSortedSetAsDilTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor

IGSortedSetAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedSetAsDilTableCursor

Overload 1

```
public:IGSortedSetAsDilTableCursor( IGSortedSetAsDilTable < Element,
                                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedSetAsDilTableCursor < Element , ElementOps >&
  operator =( IGSortedSetAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedSetAsList

Variant of ISortedSetAsList with generic operations class.

Class header file: isslst.h

IGSortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsList
IVGSortedSetAsList
ISortedSetAsList

IGSortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetAsList

Constructs an IGSortedSetAsList object.

Overload 1

```
public:IGSortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGSortedSetAsList( IGSortedSetAsList < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGSortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetAsList object to another.

operator =

```
public:IGSortedSetAsList < Element , ElementOps >&
  operator =( IGSortedSetAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSetAsList object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedSetAsListTypedef

```
protected:IGSortedSetAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsList - Type Definitions

Cursor

```
typedef IGSortedSetAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedSetAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedSetAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1653	os_mark_schema_type	1653

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet	1041	numberOfOccurrences	1046
removeAllOccurrences	1046		

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1653		

Inherited Protected Data

None

IGSortedSetAsListCursor

Class header file: isslst.h

IGSortedSetAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedSetAsListCursor

IGSortedSetAsListMngCursor

IGSortedSetAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedSetAsListCursor

Overload 1

```
public:IGSortedSetAsListCursor( IGSortedSetAsList < Element,
                                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedSetAsListCursor < Element , ElementOps >&
  operator =( IGSortedSetAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedSetAsTable

Variant of ISortedSetAsTable with generic operations class.

Class header file: isstab.h

IGSortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsTable
ISortedSetAsTable
IVGSortedSetAsTable

IGSortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetAsTable

Constructs an IGSortedSetAsTable object.

Overload 1

```
public:IGSortedSetAsTable( IGSortedSetAsTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGSortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGSortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetAsTable object to another.

operator =

```
public:IGSortedSetAsTable < Element , ElementOps >&
  operator =( IGSortedSetAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGSortedSetAsTable object to this object.

Protected Type Definitions

Protected type definitions associated with this class.

IGSortedSetAsTableTypedef

```
protected:IGSortedSetAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsTable - Type Definitions

Cursor

```
typedef IGSortedSetAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGSortedSetAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISortedSetAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGSortedSetAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected type definitions associated with this class.

IGSortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGSortedSetAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1659	os_mark_schema_type	1659

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1659		

Inherited Protected Data

None

IGSortedSetAsTableCursor

Class header file: isstab.h

IGSortedSetAsTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedSetAsTableCursor

IGSortedSetAsTableMngCursor

IGSortedSetAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGSortedSetAsTableCursor

Overload 1

```
public:IGSortedSetAsTableCursor( IGSortedSetAsTable < Element,
                                 ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGSortedSetAsTableCursor < Element , ElementOps >&
  operator =( IGSortedSetAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGSortedSetAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGSortedSetCursor

Class header file: iss.h

IGSortedSetCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGSortedSetCursor

IGSortedSetMngCursor

IGSortedSetCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGSortedSetCursor

Constructs an IGSortedSetCursor object.

Overload 1

```
public:IGSortedSetCursor( IGSortedSet < Element,
                          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGSortedSetCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGSortedSetCursor object to another.

operator =

```
public:IGSortedSetCursor < Element , ElementOps >&
  operator =( IGSortedSetCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign one cursor to another.

IGSortedSetCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGStack

Variant of IStack with generic operations class.

Class header file: istk.h

IGStack - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStack
IVGStack
IStack

IGStack - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGStack

Constructs an IGStack object.

Overload 1

```
public:IGStack(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
public:IGStack(IGStack < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
protected:IGStack(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGStack object to another.

operator =

```
public:IGStack < Element , ElementOps >&
  operator =( IGStack < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGStack object to another.

Miscellaneous Members

IGStackTypedef

```
protected:IGStackTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStack - Type Definitions

Cursor

```
typedef IGStackCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGStackMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGStack < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStack - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGStack < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1667	os_mark_schema_type	1667

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1667		

Inherited Protected Data

None

IGStackAsDilTable

Variant of IStackAsDilTable with generic operations class.

Class header file: istkdil.h

IGStackAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsDilTable
IVGStackAsDilTable
IStackAsDilTable

IGStackAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGStackAsDilTable

Constructs an IGStackAsDilTable object.

Overload 1

```
public:IGStackAsDilTable( IGStackAsDilTable < Element,
                    ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGStackAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGStackAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGStackAsDilTable object to another.

operator =

```
public:IGStackAsDilTable < Element , ElementOps >&
  operator =( IGStackAsDilTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGStackAsDilTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGStackAsDilTableTypedef

```
protected:IGStackAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsDilTable - Type Definitions

Cursor

```
typedef IGStackAsDilTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGStackAsDilTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsDilTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGStackAsDilTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGStackAsDilTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1670	os_mark_schema_type	1670

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1670		

Inherited Protected Data

None

IGStackAsDilTableCursor

Class header file: istkdil.h

IGStackAsDilTableCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGStackAsDilTableCursor

IGStackAsDilTableMngCursor

IGStackAsDilTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGStackAsDilTableCursor

Overload 1

```
public:IGStackAsDilTableCursor( IGStackAsDilTable < Element,
                                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGStackAsDilTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGStackAsDilTableCursor < Element , ElementOps >&
  operator =( IGStackAsDilTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsDilTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
`ICursor`	1100	`ImplOf`	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IElementCursor`	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
`IOrderedCursor`	1810		

Inherited Protected Data

None

IGStackAsList

Variant of IStackAsList with generic operations class.

Class header file: istklst.h

IGStackAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsList
IStackAsList
IVGStackAsList

IGStackAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGStackAsList

Constructs an IGStackAsList object.

Overload 1

```
public:IGStackAsList(IGStackAsList < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGStackAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGStackAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGStackAsList object to another.

operator =

```
public:IGStackAsList < Element , ElementOps >&
  operator =( IGStackAsList < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGStackAsList object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGStackAsListTypedef

```
protected:IGStackAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsList - Type Definitions

Cursor

```
typedef IGStackAsListCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGStackAsListMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsListImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGStackAsList < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGStackAsList < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsList - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1676	os_mark_schema_type	1676

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1676		

Inherited Protected Data

None

IGStackAsListCursor

Class header file: istklst.h

IGStackAsListCursor - Hierarchy List

ICursor

IElementCursor

IOrderedCursor

IGStackAsListCursor

IGStackAsListMngCursor

IGStackAsListCursor - Member Functions and Data by Group

Constructors & Destructor

IGStackAsListCursor

Overload 1

```
public:IGStackAsListCursor( IGStackAsList < Element,
                      ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGStackAsListCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGStackAsListCursor < Element , ElementOps >&
  operator =( IGStackAsListCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsListCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGStackAsTable

Variant of IStackAsTable with generic operations class.

Class header file: istktab.h

IGStackAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAStack
        IGStackAsTable
          IVGStackAsTable

IGStackAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IGStackAsTable

Constructs an IGStackAsTable object.

Overload 1

```
public:IGStackAsTable( IGStackAsTable < Element,
                ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IGStackAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 3

```
protected:IGStackAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

This operator assigns one IGStackAsTable object to another.

operator =

```
public:IGStackAsTable < Element , ElementOps >&
  operator =( IGStackAsTable < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns one IGStackAsTable object to another.

Protected Type Definitions

Protected type definitions associated with this class.

IGStackAsTableTypedef

```
protected:IGStackAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsTable - Type Definitions

Cursor

```
typedef IGStackAsTableCursor < Element , ElementOps > Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef PureImplementation Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

MngCursor

```
typedef IGStackAsTableMngCursor < Element , ElementOps > MngCursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef ISequenceAsTableImpl PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IGStackAsTable < Element , ElementOps > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ImplOf

```
static Implementation&
  ImplOf( IGStackAsTable < Element,
          ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1681	os_mark_schema_type	1681

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
ImplOf	1681		

Inherited Protected Data

None

IGStackAsTableCursor

Class header file: istktab.h

IGStackAsTableCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGStackAsTableCursor
        IGStackAsTableMngCursor

IGStackAsTableCursor - Member Functions and Data by Group

Constructors & Destructor

IGStackAsTableCursor

Overload 1

```
public:IGStackAsTableCursor( IGStackAsTable < Element,
                        ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGStackAsTableCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGStackAsTableCursor < Element , ElementOps >&
  operator =( IGStackAsTableCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackAsTableCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGStackCursor

Class header file: istk.h

IGStackCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGStackCursor
        IGStackMngCursor

IGStackCursor - Member Functions and Data by Group

Constructors & Destructor

IGStackCursor

Overload 1

```
public:IGStackCursor(IGStack < Element, ElementOps > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IGStackCursor(ICursorImpl*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assignment Operator

operator =

```
public:IGStackCursor < Element , ElementOps >&
  operator =( IGStackCursor < Element,
              ElementOps > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IGStackCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
setToLast	1810	setToPrevious	1810

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

IOrderedCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IOrderedCursor	1810		

Inherited Protected Data

None

IGTree

IGTree is the default implementation variant with generic operations class.

Class header file: itree.h

IGTree - Hierarchy List

IMetaTree
  IATreeBase
    IATree
      IAMultiwayTree
        IGMultiwayTree
          IGTree
            ITree

IGTree - Member Functions and Data by Group

Assignment

Assign an IGTree objet to this object.

operator =

```
public:IGTree < Element , ElementOps , numOfChildren >&
  operator =( IGTree < Element,
              ElementOps,
              numOfChildren > const& tree )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IGTree objet to this object.

IGTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946
hasChild	947	numberOfChildren	947

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1399	IGMultiwayTree	1398
operator =	1398	os_mark_schema_type	1399

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree ImplOf	1398 1398	IGMultiwayTreeTypedef	1398

Inherited Protected Data

None

IHeap

IHeap is the default implementation variant, as a list, of IAHeap.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Assignment

Class header file: ihp.h

IHeap - Hierarchy List

IGHeap
  IHeap

IHeap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IHeap

Constructs an IHeap object.

Overload 1

```
public:IHeap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IHeap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IHeap < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeap - Inherited Member Functions and Data

Inherited Public Functions

IHeap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1690	os_mark_schema_type	1690

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap	1216	operator =	1217

Inherited Public Data

None

Inherited Protected Functions

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap ImplOf	1216 1218	IGHeapTypedef	1217

Inherited Protected Data

None

IHeapAsDilTable

IHeapAsDilTable is the diluted table implementation variant of IAHeap.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Assignment

Class header file: ihpdil.h

IHeapAsDilTable - Hierarchy List

IGHeapAsDilTable

IHeapAsDilTable

IHeapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IHeapAsDilTable

Constructs an IHeapAsDilTable object.

Overload 1

```
public:IHeapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IHeapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IHeapAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1692	os_mark_schema_type	1692

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable	1219	operator =	1219

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable ImplOf	1219 1220	IGHeapAsDilTableTypedef	1219

Inherited Protected Data

None

IHeapAsList

IHeapAsList is the list implementation variant of IAHeap.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Assignment

Class header file: ihplst.h

IHeapAsList - Hierarchy List

IGHeapAsList

IHeapAsList

IHeapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IHeapAsList

Constructs an IHeapAsList object.

Overload 1

```
public:IHeapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IHeapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IHeapAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsList - Inherited Member Functions and Data

Inherited Public Functions

IHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1693	os_mark_schema_type	1693

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList	1222	operator =	1223

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList ImplOf	1222 1224	IGHeapAsListTypedef	1223

Inherited Protected Data

None

IHeapAsTable

IHeapAsTable is the table implementation variant of IAHeap.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Assignment

Class header file: ihptab.h

IHeapAsTable - Hierarchy List

IGHeapAsTable

IHeapAsTable

IHeapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IHeapAsTable

Constructs an IHeapAsTable object.

Overload 1

```
public:IHeapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IHeapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IHeapAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHeapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1695	os_mark_schema_type	1695

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsTable	1226	operator =	1227

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
`IGHeapAsTable` `ImplOf`	1226 1227	`IGHeapAsTableTypedef`	1227

Inherited Protected Data

None

IInternalApplicator

Internal class that defines the interface for internal applicator objects. The redefinition of the function applyTo() defines the actions that are performed with the version of allElementsDo() that takes an applicator argument. Iteration stops when applyTo() returns false.

Class header file: iappl.h

IInternalApplicator - Hierarchy List

IInternalApplicator

IInternalApplicator - Member Functions and Data by Group

Apply To

The applyTo function.

applyTo

```
public:virtual bool applyTo(void*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal applyTo function.

IInternalApplicator - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IKeyBag

IKeyBag is the default implementation variant, as a hash table, of IAKeyBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Equality test
- Hash function

Class header file: ikb.h

IKeyBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBag
IKeyBag

IKeyBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeyBag

Constructs an IKeyBag object.

Overload 1

```
public:IKeyBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeyBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeyBag < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1698	os_mark_schema_type	1698

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBag	1231	operator =	1232

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAKeyCollection`	909		

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
`IGKeyBag` `ImplOf`	1231 1233	`IGKeyBagTypedef`	1232

Inherited Protected Data

None

IKeyBagAsDilTable

IKeyBagAsDilTable is the diluted table implementation variant of IAKeyBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Equality test

Class header file: ikbdil.h

IKeyBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsDilTable
IKeyBagAsDilTable

IKeyBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeyBagAsDilTable

Constructs an IKeyBagAsDilTable object.

Overload 1

```
public:IKeyBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeyBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeyBagAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1701	os_mark_schema_type	1701

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable	1235	operator =	1236

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable ImplOf	1235 1236	IGKeyBagAsDilTableTypedef	1236

Inherited Protected Data

None

IKeyBagAsHshTable

IKeyBagAsHshTable is the hash table implementation variant of IAKeyBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type

- Equality test
- Hash function

Class header file: ikbhsh.h

IKeyBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsHshTable
IKeyBagAsHshTable

IKeyBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeyBagAsHshTable

Constructs an IKeyBagAsHshTable object.

Overload 1

```
public:IKeyBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeyBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeyBagAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1704	os_mark_schema_type	1705

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable	1240	operator =	1241

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable	1240	IGKeyBagAsHshTableTypedef	1241
ImplOf	1242		

Inherited Protected Data

None

IKeyBagAsList

IKeyBagAsList is the list implementation variant of IAKeyBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Equality test

Class header file: ikblst.h

IKeyBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsList
IKeyBagAsList

IKeyBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeyBagAsList

Constructs an IKeyBagAsList object.

Overload 1

```
public:IKeyBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeyBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeyBagAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1708	os_mark_schema_type	1708

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList	1246	operator =	1246

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList ImplOf	1246 1247	IGKeyBagAsListTypedef	1246

Inherited Protected Data

None

IKeyBagAsTable

IKeyBagAsTable is the table implementation variant of IAKeyBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Equality test

Class header file: ikbtab.h

IKeyBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsTable
IKeyBagAsTable

IKeyBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeyBagAsTable

Constructs an IKeyBagAsTable object.

Overload 1

```
public:IKeyBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeyBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeyBagAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeyBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1711	os_mark_schema_type	1711

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable	1251	operator =	1252

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable	1251	IGKeyBagAsTableTypedef	1252
ImplOf	1253		

Inherited Protected Data

None

IKeySet

IKeySet is the default implementation variant of IAKeySet.

The default implementation of the class IKeySet requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type

Ordering relation

Class header file: iks.h

IKeySet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySet
IKeySet

IKeySet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySet

Constructs an IKeySet object.

Overload 1

```
public:IKeySet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySet < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

IKeySet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1715	os_mark_schema_type	1715

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	operator =	1259

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	IGKeySetTypedef	1258
ImplOf	1260		

Inherited Protected Data

None

IKeySetAsAvlTree

IKeySetAsAvlTree is the AVL tree implementation variant of IAKeySet.

The implementation of the class IKeySetAsAvlTree requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type

Ordering relation

Class header file: iksavl.h

IKeySetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsAvlTree

IKeySetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsAvlTree

Constructs an IKeySetAsAvlTree object.

Overload 1

```
public:IKeySetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1718	os_mark_schema_type	1718

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree	1262	operator =	1263

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree	1262	IGKeySetAsAvlTreeTypedef	1263
ImplOf	1264		

Inherited Protected Data

None

IKeySetAsBstTree

IKeySetAsBstTree is the B* tree implementation variant of IAKeySet.

The implementation of the class IKeySetAsBstTree requires the following element and key-type functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksbst.h

IKeySetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsBstTree
IKeySetAsBstTree

IKeySetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsBstTree

Constructs an IKeySetAsBstTree object.

Overload 1

```
public:IKeySetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1722	os_mark_schema_type	1722

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree	1268	operator =	1268

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet numberOfDifferentKeys removeAllElementsWithKey	917 918 918	locateNextElementWithKey numberOfElementsWithKey setToNextWithDifferentKey	918 918 919

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree ImplOf	1268 1269	IGKeySetAsBstTreeTypedef	1268

Inherited Protected Data

None

IKeySetAsDilTable

IKeySetAsDilTable is the diluted table implementation variant of IAKeySet.

The implementation class IKeySetAsDilTable requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksdil.h

IKeySetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsDilTable
IKeySetAsDilTable

IKeySetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsDilTable

Constructs an IKeySetAsDilTable object.

Overload 1

```
public:IKeySetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1725	os_mark_schema_type	1725

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	operator =	1274

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	IGKeySetAsDilTableTypedef	1274
ImplOf	1275		

Inherited Protected Data

None

IKeySetAsHshTable

IKeySetAsHshTable is the hash table implementation variant of IAKeySet.

The implementation class IKeySetAsHshTable requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Equality test
- Hash function

Class header file: ikshsh.h

IKeySetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsHshTable
IKeySetAsHshTable

IKeySetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsHshTable

Constructs an IKeySetAsHshTable object.

Overload 1

```
public:IKeySetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1729	os_mark_schema_type	1729

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable	1279	operator =	1279

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable ImplOf	1279 1280	IGKeySetAsHshTableTypedef	1279

Inherited Protected Data

None

IKeySetAsList

IKeySetAsList is the list implementation variant of IAKeySet.

The implementation class IKeySetAsList requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikslst.h

IKeySetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsList
IKeySetAsList

IKeySetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsList

Constructs an IKeySetAsList object.

Overload 1

```
public:IKeySetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1732	os_mark_schema_type	1732

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	operator =	1285

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	IGKeySetAsListTypedef	1285
ImplOf	1286		

Inherited Protected Data

None

IKeySetAsTable

IKeySetAsTable is the table implementation variant of IAKeySet.

The implementation class IKeySetAsTable requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikstab.h

IKeySetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsTable
IKeySetAsTable

IKeySetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySetAsTable

Constructs an IKeySetAsTable object.

Overload 1

```
public:IKeySetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySetAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1735	os_mark_schema_type	1735

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable	1290	operator =	1290

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet numberOfDifferentKeys removeAllElementsWithKey	917 918 918	locateNextElementWithKey numberOfElementsWithKey setToNextWithDifferentKey	918 918 919

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable ImplOf	1290 1291	IGKeySetAsTableTypedef	1290

Inherited Protected Data

None

IKeySortedBag

IKeySortedBag is the default implementation variant, as a list, of IAKeySortedBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksb.h

IKeySortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBag
IKeySortedBag

IKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedBag

Constructs an IKeySortedBag object.

Overload 1

```
public:IKeySortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedBag < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1739	os_mark_schema_type	1739

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag	1297	operator =	1297

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag ImplOf	1297 1298	IGKeySortedBagTypedef	1296

Inherited Protected Data

None

IKeySortedBagAsDilTable

IKeySortedBagAsDilTable is the diluted table implementation variant of IAKeySortedBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksbdil.h

IKeySortedBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsDilTable
IKeySortedBagAsDilTable

IKeySortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedBagAsDilTable

Constructs an IKeySortedBagAsDilTable object.

Overload 1

```
public:IKeySortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedBagAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag IAKeySortedBag	921 922	compare	923

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1742	os_mark_schema_type	1743

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable	1301	operator =	1302

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable ImplOf	1301 1303	IGKeySortedBagAsDilTableTypedef	1302

Inherited Protected Data

None

IKeySortedBagAsList

IKeySortedBagAsList is the list implementation variant of IAKeySortedBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksblst.h

IKeySortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsList
IKeySortedBagAsList

IKeySortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedBagAsList

Constructs an IKeySortedBagAsList object.

Overload 1

```
public:IKeySortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedBagAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1746	os_mark_schema_type	1746

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList	1307	operator =	1308

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList ImplOf	1307 1309	IGKeySortedBagAsListTypedef	1308

Inherited Protected Data

None

IKeySortedBagAsTable

IKeySortedBagAsTable is the table implementation variant of IAKeySortedBag.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksbtab.h

IKeySortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsTable
IKeySortedBagAsTable

IKeySortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedBagAsTable

Constructs an IKeySortedBagAsTable object.

Overload 1

```
public:IKeySortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedBagAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1750	os_mark_schema_type	1750

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable	1313	operator =	1314

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable ImplOf	1313 1315	IGKeySortedBagAsTableTypedef	1314

Inherited Protected Data

None

IKeySortedSet

IKeySortedSet is the default implementation variant, as an AVL tree, of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikss.h

IKeySortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSet
IKeySortedSet

IKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedSet

Constructs an IKeySortedSet object.

Overload 1

```
public:IKeySortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSet < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAllElementsWithKey replaceElementWithKey	913 914	removeElementWithKey	914

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1754	os_mark_schema_type	1754

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet	1321	operator =	1322

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet	1321	IGKeySortedSetTypedef	1322
ImplOf	1323		

Inherited Protected Data

None

IKeySortedSetAsAvlTree

IKeySortedSetAsAvlTree is the AVL tree implementation variant of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikssavl.h

IKeySortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsAvlTree
IKeySortedSetAsAvlTree

IKeySortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedSetAsAvlTree

Constructs an IKeySortedSetAsAvlTree object.

Overload 1

```
public:IKeySortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSetAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1757	os_mark_schema_type	1757

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree	1326	operator =	1326

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree ImplOf	1326 1327	IGKeySortedSetAsAvlTreeTypedef	1326

Inherited Protected Data

None

IKeySortedSetAsBstTree

IKeySortedSetAsBstTree is the B* tree implementation variant of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikssbst.h

IKeySortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsBstTree
IKeySortedSetAsBstTree

IKeySortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedSetAsBstTree

Constructs an IKeySortedSetAsBstTree object.

Overload 1

```
public:IKeySortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSetAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1761	os_mark_schema_type	1761

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree	1332	operator =	1332

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree ImplOf	1332 1333	IGKeySortedSetAsBstTreeTypedef	1332

Inherited Protected Data

None

IKeySortedSetAsDilTable

IKeySortedSetAsDilTable is the diluted table implementation variant of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ikssdil.h

IKeySortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsDilTable
IKeySortedSetAsDilTable

IKeySortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IKeySortedSetAsDilTable

Constructs an IKeySortedSetAsDilTable object.

Overload 1

```
public:IKeySortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSetAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1765	os_mark_schema_type	1765

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable	1338	operator =	1338

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable ImplOf	1338 1339	IGKeySortedSetAsDilTableTypedef	1338

Inherited Protected Data

None

IKeySortedSetAsList

IKeySortedSetAsList is the list implementation variant of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksslst.h

IKeySortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsList
IKeySortedSetAsList

IKeySortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IKeySortedSetAsList

Constructs an IKeySortedSetAsList object.

Overload 1

```
public:IKeySortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSetAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1769	os_mark_schema_type	1769

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList	1344	operator =	1344

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList ImplOf	1344 1345	IGKeySortedSetAsListTypedef	1344

Inherited Protected Data

None

IKeySortedSetAsTable

IKeySortedSetAsTable is the table implementation variant of IAKeySortedSet.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: iksstab.h

IKeySortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsTable
IKeySortedSetAsTable

IKeySortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IKeySortedSetAsTable

Constructs an IKeySortedSetAsTable object.

Overload 1

```
public:IKeySortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IKeySortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IKeySortedSetAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKeySortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1773	os_mark_schema_type	1773

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable	1350	operator =	1350

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable ImplOf	1350 1351	IGKeySortedSetAsTableTypedef	1350

Inherited Protected Data

None

IMap

IMap is the default implementation variant, as an AVL tree, of IAHeap.

The default implementation of the class IMap requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Ordering relation

Class header file: imap.h

IMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMap
IMap

IMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMap

Constructs an IMap object.

Overload 1

```
public:IMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMap < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1777	os_mark_schema_type	1777

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGMap	1357	operator =	1358

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGMap	1357	IGMapTypedef	1358
ImplOf	1359		

Inherited Protected Data

None

IMapAsAvlTree

IMapAsAvlTree is the AVL tree implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Ordering relation

Class header file: imapavl.h

IMapAsAvlTree - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection

IAEqualityKeyCollection
IAMap
IGMapAsAvlTree
IMapAsAvlTree

IMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMapAsAvlTree

Constructs an IMapAsAvlTree object.

Overload 1

```
public:IMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	938	unionWith	938

IMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1780	os_mark_schema_type	1780

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree	1362	operator =	1362

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap locateNextElementWithKey numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	934 939 940 940 941	locateNext numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	939 940 940 941

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree ImplOf	1362 1363	IGMapAsAvlTreeTypedef	1362

Inherited Protected Data

None

IMapAsBstTree

IMapAsBstTree is the B* tree implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Ordering relation

Class header file: imapbst.h

IMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsBstTree
IMapAsBstTree

IMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMapAsBstTree

Constructs an IMapAsBstTree object.

Overload 1

```
public:IMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1784	os_mark_schema_type	1784

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsBstTree	1367	operator =	1368

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	941		

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsBstTree ImplOf	1367 1369	IGMapAsBstTreeTypedef	1368

Inherited Protected Data

None

IMapAsDilTable

IMapAsDilTable is the diluted table implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Ordering relation

Class header file: imapdil.h

IMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsDilTable
IMapAsDilTable

IMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IMapAsDilTable

Constructs an IMapAsDilTable object.

Overload 1

```
public:IMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1788	os_mark_schema_type	1788

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsDilTable	1373	operator =	1374

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
`IGMapAsDilTable` `ImplOf`	1373 1374	`IGMapAsDilTableTypedef`	1374

Inherited Protected Data

None

IMapAsHshTable

IMapAsHshTable is the hash table implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Equality test
- Hash function

Class header file: imaphsh.h

IMapAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsHshTable
IMapAsHshTable

IMapAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMapAsHshTable

Constructs an IMapAsHshTable object.

Overload 1

```
public:IMapAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1792	os_mark_schema_type	1792

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable	1379	operator =	1379

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable	1379	IGMapAsHshTableTypedef	1379
ImplOf	1380		

Inherited Protected Data

None

IMapAsList

IMapAsList is the list implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Ordering relation

Class header file: imaplst.h

IMapAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsList
IMapAsList

IMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMapAsList

Constructs an IMapAsList object.

Overload 1

```
public:IMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1796	os_mark_schema_type	1796

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList	1385	operator =	1385

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList	1385	IGMapAsListTypedef	1385
ImplOf	1386		

Inherited Protected Data

None

IMapAsTable

IMapAsTable is the table implementation variant of IAMap.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test

- Key access

Key Type
- Ordering relation

Class header file: imaptab.h

IMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsTable
IMapAsTable

IMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IMapAsTable

Constructs an IMapAsTable object.

Overload 1

```
public:IMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMapAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1800	os_mark_schema_type	1800

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable	1391	operator =	1391

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable	1391	IGMapAsTableTypedef	1391
ImplOf	1392		

Inherited Protected Data

None

IMngElemPointer

Use objects of the IMngPointer class as managed element pointers for collections. Managed element pointers route the operations on the pointers to the referenced elements, and keep a reference count for each referenced element. When the last managed pointer to the element is destructed, the element is automatically deleted.

Class header file: iptr.h

IMngElemPointer - Hierarchy List

IMngPointer

IMngElemPointer

IMngElemPointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct a managed elemented pointer object.

IMngElemPointer

Constructs a pointer object of the indicated type from a given C++ pointer. The reference count of the referenced element is set to 1.

Overload 1

```
public:IMngElemPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IMngElemPointer(Element*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IMngElemPointer(Element*, IExplicitInit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer object of the indicated type from a given C++ pointer. The reference count of the referenced element is set to 1.

Overload 4

```
public:IMngElemPointer(Element const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Operators

Use these operators to access the element being managed by this object.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object to which the pointer refers.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a C pointer to the object to which the pointer refers.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicitly converts this pointer to a C++ pointer.

IMngElemPointer - Inherited Member Functions and Data

Inherited Public Functions

IMngPointer			
Definition	**Page Number**	**Definition**	**Page Number**
~IMngPointer operator =	1804 1804	IMngPointer	1804

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMngPointer

Use objects of the IMngPointer class as managed pointers for collections. Managed pointers keep a reference count for each referenced element. When the last managed pointer to the element is destructed, the element is automatically deleted.

Class header file: iptr.h

IMngPointer - Hierarchy List

IMngPointer

IMngElemPointer

IMngPointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct a managed pointer object.

~IMngPointer

```
public:~IMngPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs the pointer and decrements the reference count of the referenced element. If the reference count is 0, the referenced element is deleted.

IMngPointer

Constructs an IMngPointer object.

Overload 1

```
public:IMngPointer(Element*, IExplicitInit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type from a given C++ pointer. The reference count of the referenced element is set to 1.

Overload 2

```
public:IMngPointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a pointer of the indicated type and initializes it with NULL.

Overload 3

```
public:IMngPointer(IMngPointer < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a new pointer and initializes it with the given pointer. The reference count of the referenced element is incremented by 1.

operator =

```
public:IMngPointer < Element >&
  operator =( IMngPointer < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the given pointer to this pointer. The reference count of the referenced element is incremented and the reference count of the previously referenced element is decremented.

Operators

Use these operators to access the element being managed by this object.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object to which the pointer refers.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a C pointer to the object to which the pointer refers.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicitly converts this pointer to a C++ pointer.

IMngPointer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMultiwayTree

The default implementation of IMultiwayTree requires the following element functions:

Element Type

- Copy constructor
- Destructor
- Assignment

Class header file: imwt.h

IMultiwayTree - Hierarchy List

IMetaTree

IATreeBase

IATree
IAMultiwayTree
IGMultiwayTree
IMultiwayTree

IMultiwayTree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IMultiwayTree

Constructs an IMultiwayTree object.

Overload 1

```
public:IMultiwayTree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree. The tree is initially empty; that is, it does not contain any nodes.

Overload 2

```
protected:IMultiwayTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a tree by copying all elements from the given tree.

Exception

IOutOfMemory

IMultiwayTree - Type Definitions

Guard

```
typedef ITreeCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IMultiwayTree < numOfChildren , Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

IMultiwayTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMultiwayTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946
hasChild	947	numberOfChildren	947

IMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1806	os_mark_schema_type	1806

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree	1398	operator =	1398

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree	1398	IGMultiwayTreeTypedef	1398
ImplOf	1398		

Inherited Protected Data

None

IOrderedCursor

Cursors of ordered collections are derived from IOrderedCursor.

Because the cursor classes are all abstract classes, no objects of type IOrderedCursor, IElementCursor, or ICursor can be declared. You can obtain cursor objects by using the collection member IACollection::newCursor() or by defining a cursor of a specific collection cursor class. The newCursor() member creates a cursor of the collection to which it is applied.

The newCursor() member returns a pointer to the newly created cursor object.

Each cursor object is associated with a collection object. A cursor function merely calls the corresponding function for this collection. For example, cursor.setToFirst() is the same as collection.setToFirst(cursor), where collection is the object associated with cursor.

Class header file: iaorder.h

IOrderedCursor - Hierarchy List

ICursor
  IElementCursor
    IOrderedCursor
      IGKeySortedBagAsDilTableCursor
      IGSortedSetAsAvlTreeCursor
      IGDequeAsTableCursor
      IGSequenceAsListCursor
      IGKeySortedSetAsBstTreeCursor
      IGSortedMapAsListCursor
      IGSortedSetAsBstTreeCursor
      IGSortedSetAsListCursor
      IGSortedBagAsAvlTreeCursor
      IGEqualitySequenceCursor
      IGSortedBagCursor
      IGKeySortedBagAsListCursor
      IGSortedSetAsTableCursor
      IGSortedBagAsBstTreeCursor
      IGKeySortedSetAsDilTableCursor
      IGSortedMapCursor
      IGDequeCursor
      IGSortedBagAsListCursor
      IGSortedMapAsTableCursor
      IGPriorityQueueAsDilTableCursor
      IGStackCursor
      IGKeySortedSetAsTableCursor
      IGQueueAsListCursor
      IGQueueCursor
      IGSortedBagAsDilTableCursor
      IGSortedBagAsTableCursor
      IGSequenceCursor
      IGKeySortedBagAsTableCursor
      IGQueueAsDilTableCursor
      IGKeySortedSetAsListCursor
      IGKeySortedBagCursor
      IGPriorityQueueAsListCursor
      IGQueueAsTableCursor
      IGSortedRelationAsDilTableCursor
      IGEqualitySequenceAsListCursor

IGSortedRelationCursor
IGSortedMapAsBstTreeCursor
IGStackAsListCursor
IGDequeAsDilTableCursor
IGSortedMapAsAvlTreeCursor
IGPriorityQueueCursor
IGEqualitySequenceAsDilTableCursor
IGEqualitySequenceAsTableCursor
IGKeySortedSetAsAvlTreeCursor
IGStackAsDilTableCursor
IGKeySortedSetCursor
IGPriorityQueueAsTableCursor
IGStackAsTableCursor
IGSortedRelationAsTableCursor
IGDequeAsListCursor
IGSequenceAsDilTableCursor
IGSortedSetCursor
IGSequenceAsTableCursor
IGSortedMapAsDilTableCursor
IGSortedSetAsDilTableCursor

IOrderedCursor - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IOrderedCursor

```
protected:IOrderedCursor(Implementation*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IOrderCursor object.

Iteration

These member functions are used for cursor iteration.

setToLast

```
public:bool setToLast()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last element of the associated collection in iteration order. Invalidates the cursor if the collection is empty (no element exists).

Return Value

Returns true if the associated collection was not empty.

setToPrevious

```
public:bool setToPrevious()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous element of the associated collection in iteration order. Invalidates the cursor if no such element exists.

Return Value

Returns true if a previous element exists.

Precondition

The cursor must point to an element of the associated collection.

Exception
- ICursorInvalidException

IOrderedCursor - Inherited Member Functions and Data

Inherited Public Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
~ICursor	1100	copy	1100
invalidate	1101	isInBetween	1101
isMngCursor	1101	isValid	1101
operator !=	1101	operator =	1100
operator ==	1101	setToFirst	1101
setToNext	1101		

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
element	1116		

Inherited Public Data

None

Inherited Protected Functions

ICursor			
Definition	**Page Number**	**Definition**	**Page Number**
ICursor	1100	ImplOf	1100

IElementCursor			
Definition	**Page Number**	**Definition**	**Page Number**
IElementCursor	1116		

Inherited Protected Data

None

IPriorityQueue

IPriorityQueue is the default implementation variant, as a list, of IAPriorityQueue.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ipqu.h

IPriorityQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueue
IPriorityQueue

IPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IPriorityQueue

Constructs an IPriorityQueue object.

Overload 1

```
public:IPriorityQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IPriorityQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IPriorityQueue < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1813	os_mark_schema_type	1813

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue	1403	operator =	1403

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue ImplOf	1403 1404	IGPriorityQueueTypedef	1403

Inherited Protected Data

None

IPriorityQueueAsDilTable

IPriorityQueueAsDilTable is the diluted table implementation variant of IAPriorityQueue.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ipqudil.h

IPriorityQueueAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsDilTable
IPriorityQueueAsDilTable

IPriorityQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IPriorityQueueAsDilTable

Constructs an IPriorityQueueAsDilTable object.

Overload 1

```
public:IPriorityQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IPriorityQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IPriorityQueueAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1816	os_mark_schema_type	1816

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable	1407	operator =	1407

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable	1407	IGPriorityQueueAsDilTableTypedef	1407
ImplOf	1408		

Inherited Protected Data

None

IPriorityQueueAsList

IPriorityQueueAsList is the list implementation variant of IAPriorityQueue.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ipqulst.h

IPriorityQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsList
IPriorityQueueAsList

IPriorityQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IPriorityQueueAsList

Constructs an IPriorityQueueAsList object.

Overload 1

```
public:IPriorityQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IPriorityQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IPriorityQueueAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1819	os_mark_schema_type	1819

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList	1412	operator =	1413

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList	1412	IGPriorityQueueAsListTypedef	1413
ImplOf	1414		

Inherited Protected Data

None

IPriorityQueueAsTable

IPriorityQueueAsTable is the table implementation variant of IAPriorityQueue.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access

Key Type
- Ordering relation

Class header file: ipqutab.h

IPriorityQueueAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsTable
IPriorityQueueAsTable

IPriorityQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IPriorityQueueAsTable

Constructs an IPriorityQueueAsTable object.

Overload 1

```
public:IPriorityQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IPriorityQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IPriorityQueueAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPriorityQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1823	os_mark_schema_type	1823

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable	1418	operator =	1418

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable ImplOf	1418 1419	IGPriorityQueueAsTableTypedef	1419

Inherited Protected Data

None

IQueue

IQueue is the default implementation variant, as a list, of IAQueue.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iqu.h

IQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueue
IQueue

IQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IQueue

Constructs an IQueue object.

Overload 1

```
public:IQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IQueue < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueue - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1826	os_mark_schema_type	1826

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue	1425	operator =	1426

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue ImplOf	1425 1426	IGQueueTypedef	1426

Inherited Protected Data

None

IQueueAsDilTable

IQueueAsDilTable is the diluted table implementation variant of IAQueue.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iqudil.h

IQueueAsDilTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAQueue
        IGQueueAsDilTable
          IQueueAsDilTable

IQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IQueueAsDilTable

Constructs an IQueueAsDilTable object.

Overload 1

```
public:IQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IQueueAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1829	os_mark_schema_type	1829

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable	1429	operator =	1429

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable ImplOf	1429 1430	IGQueueAsDilTableTypedef	1429

Inherited Protected Data

None

IQueueAsList

IQueueAsList is the list implementation variant of IAQueue.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iqulst.h

IQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsList
IQueueAsList

IQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IQueueAsList

Constructs an IQueueAsList object.

Overload 1

```
public:IQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IQueueAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1832	os_mark_schema_type	1832

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList	1434	operator =	1435

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList	1434	IGQueueAsListTypedef	1435
ImplOf	1436		

Inherited Protected Data

None

IQueueAsTable

IQueueAsTable is the table implementation variant of IAQueue.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iqutab.h

IQueueAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAQueue
        IGQueueAsTable

IQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IQueueAsTable

Constructs an IQueueAsTable object.

Overload 1

```
public:IQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IQueueAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1835	os_mark_schema_type	1835

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	operator =	1440

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	IGQueueAsTableTypedef	1440
ImplOf	1441		

Inherited Protected Data

None

IRBag

Class header file: irbag.h

IRBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IABag

IRBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRBag

```
public:IRBag(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRBag object.

Assignment Operator

This operator assigns one IRBag object to another.

operator =

```
public:IRBag < Element , Base >&
  operator =( IRBag < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRBag object to this object.

IRBag - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRBag - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

Inherited Protected Data

None

IRDeque

Class header file: irdeque.h

IRDeque - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IRDeque

IRDeque - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRDeque

```
public:IRDeque(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRDeque object.

Assignment Operator

This operator assigns one IRDeque object to another.

operator =

```
public:IRDeque < Element , Base >&
  operator =( IRDeque < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRDeque object to this object.

IRDeque - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRDeque - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IRelation

IRelation is the default implementation variant of IARelation.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality test
- Key access

Key Type
- Equality test
- Hash function

Class header file: irel.h

IRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelation
IRelation

IRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRelation

Constructs an IRelation object.

Overload 1

```
public:IRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IRelation < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1843	os_mark_schema_type	1843

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation	1447	operator =	1447

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation ImplOf	1447 1448	IGRelationTypedef	1448

Inherited Protected Data

None

IRelationAsDilTable

Class header file: ireldil.h

IRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsDilTable
IRelationAsDilTable

IRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRelationAsDilTable

Constructs an IRelationAsDilTable object.

Overload 1

```
public:IRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IRelationAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1847	os_mark_schema_type	1847

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable	1451	operator =	1452

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable	1451	IGRelationAsDilTableTypedef	1452
ImplOf	1452		

Inherited Protected Data

None

IRelationAsHshTable

Class header file: irelhsh.h

IRelationAsHshTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IARelation
          - IGRelationAsHshTable
            - **IRelationAsHshTable**

IRelationAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRelationAsHshTable

Constructs an IRelationAsHshTable object.

Overload 1

```
public:IRelationAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IRelationAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IRelationAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1850	os_mark_schema_type	1850

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable	1457	operator =	1457

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable ImplOf	1457 1458	IGRelationAsHshTableTypedef	1457

Inherited Protected Data

None

IRelationAsList

Class header file: irellst.h

IRelationAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsList
IRelationAsList

IRelationAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRelationAsList

Constructs an IRelationAsList object.

Overload 1

```
public:IRelationAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IRelationAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IRelationAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1854	os_mark_schema_type	1854

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList	1462	operator =	1463

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList	1462	IGRelationAsListTypedef	1463
ImplOf	1464		

Inherited Protected Data

None

IRelationAsTable

Class header file: ireltab.h

IRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsTable
IRelationAsTable

IRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRelationAsTable

Constructs an IRelationAsTable object.

Overload 1

```
public:IRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IRelationAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1857	os_mark_schema_type	1857

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable	1468	operator =	1468

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable ImplOf	1468 1469	IGRelationAsTableTypedef	1469

Inherited Protected Data

None

IREqualitySequence

Class header file: ireqseq.h

IREqualitySequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IREqualitySequence

IREqualitySequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IREqualitySequence

```
public:IREqualitySequence(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IREqualitySequence object.

Assignment Operator

This operator assigns one IREqualitySequence object to another.

operator =

```
public:IREqualitySequence < Element , Base >&
  operator =( IREqualitySequence < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IREqualitySequence object to this object.

IREqualitySequence - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IREqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

Inherited Protected Data

None

IRestrictedAccessCollectionGuard

Use objects of the IRestrictedAccessCollectionGuard class to ensure the thread safety of a collection in a multithreaded program.

The guard object makes sure that only one thread at a time accesses a collection, by locking the collection and preventing other threads from accessing it.

The guard object must be destroyed prior to the destruction of the corresponding collection.

Class header file: iarstrct.h

IRestrictedAccessCollectionGuard - Hierarchy List

IRestrictedAccessCollectionGuard

IRestrictedAccessCollectionGuard - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct objects of this class.

~IRestrictedAccessCollectionGuard

```
public:~IRestrictedAccessCollectionGuard()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Unlocks the collection specified within the constructor of this guard.

IRestrictedAccessCollectionGuard

```
public:IRestrictedAccessCollectionGuard( IARestrictedAccessCollection
                                    < Element >& , long timeout = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a restricted access collection guard.

The constructor takes the collection object to be locked and an optional timeout value as parameters. The timeout value is specified in milliseconds. If a lock request cannot be resolved within the specified range of time, an exception is thrown. The timeout value defaults to -1. On POSIX systems this parameter is ignored.

Miscellaneous Members

IRestrictedAccessCollectionGuard

```
public:IRestrictedAccessCollectionGuard( IARestrictedAccessCollection < Element >&,
                                    ioc::numeric_t timeout = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a restricted access collection guard.

IRestrictedAccessCollectionGuard - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IRHeap

Class header file: irheap.h

IRHeap - Hierarchy List

IRHeap

IRHeap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRHeap

```
public:IRHeap(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRHeap object.

Assignment Operator

This operator assigns one IRHeap object to another.

operator =

```
public:IRHeap < Element , Base >&
  operator =( IRHeap < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRHeap object to this object.

IRHeap - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRHeap - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IRKeyBag

Class header file: irkeybag.h

IRKeyBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IRKeyBag

IRKeyBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRKeyBag

```
public:IRKeyBag(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRKeyBag object.

Assignment Operator

This operator assigns one IRKeyBag object to another.

operator =

```
public:IRKeyBag < Element , Key , Base >&
  operator =( IRKeyBag < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRKeyBag object to this object.

IRKeyBag - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

Inherited Protected Data

None

IRKeySet

Class header file: irkeyset.h

IRKeySet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IRKeySet

IRKeySet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRKeySet

```
public:IRKeySet(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRKeySet object.

Assignment Operator

This operator assigns one IRKeySet object to another.

operator =

```
public:IRKeySet < Element , Key , Base >&
  operator =( IRKeySet < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRKeySet object to this object.

IRKeySet - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

Inherited Protected Data

None

IRKeySortedBag

Class header file: irksbag.h

IRKeySortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IRKeySortedBag

IRKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRKeySortedBag

```
public:IRKeySortedBag(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRKeySortedBag object.

Assignment Operator

This operator assigns one IRKeySortedBag object to another.

operator =

```
public:IRKeySortedBag < Element , Key , Base >&
  operator =( IRKeySortedBag < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRKeySortedBag object to this object.

IRKeySortedBag - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag IAKeySortedBag	921 922	compare	923

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

Inherited Protected Data

None

IRKeySortedSet

Class header file: irksset.h

IRKeySortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IRKeySortedSet

IRKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRKeySortedSet

```
public:IRKeySortedSet(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRKeySortedSet object.

Assignment Operator

This operator assigns one IRKeySortedSet object to another.

operator =

```
public:IRKeySortedSet < Element , Key , Base >&
  operator =( IRKeySortedSet < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRKeySortedSet object to this object.

IRKeySortedSet - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

Inherited Protected Data

None

IRMap

Class header file: irmap.h

IRMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IRMap

IRMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRMap

```
public:IRMap(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRMap object.

Assignment Operator

This operator assigns one IRMap object to another.

operator =

```
public:IRMap < Element , Key , Base >&
  operator =( IRMap < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRMap object to this object.

IRMap - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

Inherited Protected Data

None

IRPriorityQueue

Class header file: irprioqu.h

IRPriorityQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IRPriorityQueue

IRPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRPriorityQueue

```
public:IRPriorityQueue(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRPriorityQueue object.

Assignment Operator

This operator assigns one IRPriorityQueue object to another.

operator =

```
public:IRPriorityQueue < Element , Key , Base >&
  operator =( IRPriorityQueue < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRPriorityQueue object to this object.

IRPriorityQueue - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IRQueue

Class header file: irqueue.h

IRQueue - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IAQueue

IRQueue

IRQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRQueue

```
public:IRQueue(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRQueue object.

Assignment Operator

This operator assigns one IRQueue object to another.

operator =

```
public:IRQueue < Element , Base >&
  operator =( IRQueue < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRQueue object to this object.

IRQueue - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRQueue - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

Inherited Protected Data

None

IRRelation

Class header file: irrel.h

IRRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IRRelation

IRRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRRelation

```
public:IRRelation(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRRelation object.

Assignment Operator

This operator assigns one IRRelation object to another.

operator =

```
public:IRRelation < Element , Key , Base >&
  operator =( IRRelation < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRRelation object to this object.

IRRelation - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	973	unionWith	975

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

Inherited Protected Data

None

IRSequence

Class header file: irseq.h

IRSequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IRSequence

IRSequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSequence

```
public:IRSequence(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSequence object.

Assignment Operator

This operator assigns one IRSequence object to another.

operator =

```
public:IRSequence < Element , Base >&
  operator =( IRSequence < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSequence object to this object.

IRSequence - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

Inherited Protected Data

None

IRSet

Class header file: irset.h

IRSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IRSet

IRSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSet

```
public:IRSet(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSet object.

Assignment Operator

This operator assigns one IRSet object to another.

operator =

```
public:IRSet < Element , Base >&
  operator =( IRSet < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSet object to this object.

IRSet - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

Inherited Protected Data

None

IRSortedBag

Class header file: irsrtbag.h

IRSortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IRSortedBag

IRSortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSortedBag

```
public:IRSortedBag(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSortedBag object.

Assignment Operator

This operator assigns one IRSortedBag object to another.

operator =

```
public:IRSortedBag < Element , Base >&
  operator =( IRSortedBag < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSortedBag object to this object.

IRSortedBag - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
unionWith	1017		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

Inherited Protected Data

None

IRSortedMap

Class header file: irsrtmap.h

IRSortedMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IRSortedMap

IRSortedMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSortedMap

```
public:IRSortedMap(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSortedMap object.

Assignment Operator

This operator assigns one IRSortedMap object to another.

operator =

```
public:IRSortedMap < Element , Key , Base >&
  operator =( IRSortedMap < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSortedMap object to this object.

IRSortedMap - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

Inherited Protected Data

None

IRSortedRelation

Class header file: irsrtrel.h

IRSortedRelation - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection
        IAEqualityKeySortedCollection

IRSortedRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSortedRelation

```
public:IRSortedRelation(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSortedRelation object.

Assignment Operator

This operator assigns one IRSortedRelation object to another.

operator =

```
public:IRSortedRelation < Element , Key , Base >&
  operator =( IRSortedRelation < Element,
              Key, Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSortedRelation object to this object.

IRSortedRelation - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

Inherited Protected Data

None

IRSortedSet

Class header file: irsrtset.h

IRSortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IRSortedSet

IRSortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRSortedSet

```
public:IRSortedSet(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRSortedSet object.

Assignment Operator

This operator assigns one IRSortedSet object to another.

operator =

```
public:IRSortedSet < Element , Base >&
  operator =( IRSortedSet < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRSortedSet object to this object.

IRSortedSet - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRSortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

Inherited Protected Data

None

IRStack

Class header file: irstack.h

IRStack - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IAStack

IRStack

IRStack - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IRStack

```
public:IRStack(Base&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IRStack object.

Assignment Operator

This operator assigns one IRStack object to another.

operator =

```
public:IRStack < Element , Base >&
  operator =( IRStack < Element,
              Base > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign an IRStack object to this object.

IRStack - Type Definitions

Cursor

```
typedef TYPENAME Base::Cursor Cursor
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

Inherited Protected Data

None

ISequence

ISequence is the default implementation variant, as a list, of IASequence.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iseq.h

ISequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequence
ISequence

ISequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISequence

Constructs an ISequence object.

Overload 1

```
public:ISequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISequence < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

ISequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1904	os_mark_schema_type	1904

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence	1475	operator =	1476

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence ImplOf	1475 1476	IGSequenceTypedef	1476

Inherited Protected Data

None

ISequenceAsDilTable

ISequenceAsDilTable is the diluted table implementation variant of IASequence.

The implementation of this class requires the following element functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment

Class header file: iseqdil.h

ISequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsDilTable
ISequenceAsDilTable

ISequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISequenceAsDilTable

Constructs an ISequenceAsDilTable object.

Overload 1

```
public:ISequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISequenceAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

ISequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1908	os_mark_schema_type	1908

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable	1479	operator =	1480

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable ImplOf	1479 1481	IGSequenceAsDilTableTypedef	1480

Inherited Protected Data

None

ISequenceAsList

ISequenceAsList is the list implementation variant of IASequence.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iseqlst.h

ISequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsList
ISequenceAsList

ISequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISequenceAsList

Constructs an ISequenceAsList object.

Overload 1

```
public:ISequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISequenceAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

ISequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1912	os_mark_schema_type	1912

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList	1485	operator =	1486

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList	1485	IGSequenceAsListTypedef	1486
ImplOf	1487		

Inherited Protected Data

None

ISequenceAsTable

ISequenceAsTable is the table implementation variant of IASequence.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: iseqtab.h

ISequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsTable
ISequenceAsTable

ISequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISequenceAsTable

Constructs an ISequenceAsTable object.

Overload 1

```
public:ISequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISequenceAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

ISequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1916	os_mark_schema_type	1916

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable	1491	operator =	1492

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable ImplOf	1491 1493	IGSequenceAsTableTypedef	1492

Inherited Protected Data

None

ISet

ISet is the default implementation variant, as an AVL tree, of IASet.

The default implementation of the class ISet requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: iset.h

ISet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSet
ISet

ISet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISet

Constructs an ISet object.

Overload 1

```
public:ISet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISet < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1919	os_mark_schema_type	1919

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet	1499	operator =	1500

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet	1499	IGSetTypedef	1500
ImplOf	1500		

Inherited Protected Data

None

ISetAsAvlTree

ISetAsAvlTree is the AVL tree implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isetavl.h

ISetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsAvlTree
ISetAsAvlTree

ISetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsAvlTree

Constructs an ISetAsAvlTree object.

Overload 1

```
public:ISetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1923	os_mark_schema_type	1923

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	operator =	1504

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	IGSetAsAvlTreeTypedef	1504
ImplOf	1504		

Inherited Protected Data

None

ISetAsBstTree

ISetAsBstTree is the B* tree implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isetbst.h

ISetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsBstTree
ISetAsBstTree

ISetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsBstTree

Constructs an ISetAsBstTree object.

Overload 1

```
public:ISetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1926	os_mark_schema_type	1926

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree	1509	operator =	1509

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree	1509	IGSetAsBstTreeTypedef	1509
ImplOf	1510		

Inherited Protected Data

None

ISetAsDilTable

ISetAsDilTable is the diluted table implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isetdil.h

ISetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsDilTable
ISetAsDilTable

ISetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsDilTable

Constructs an ISetAsDilTable object.

Overload 1

```
public:ISetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1930	os_mark_schema_type	1930

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	operator =	1514

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	IGSetAsDilTableTypedef	1515
ImplOf	1515		

Inherited Protected Data

None

ISetAsHshTable

ISetAsHshTable is the hash table implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Equality Test
- Hash function

Class header file: isethsh.h

ISetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsHshTable
ISetAsHshTable

ISetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsHshTable

Constructs an ISetAsHshTable object.

Overload 1

```
public:ISetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsHshTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1933	os_mark_schema_type	1933

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable	1519	operator =	1520

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable	1519	IGSetAsHshTableTypedef	1520
ImplOf	1521		

Inherited Protected Data

None

ISetAsList

ISetAsList is the list implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isetlst.h

ISetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsList
ISetAsList

ISetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsList

Constructs an ISetAsList object.

Overload 1

```
public:ISetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1936	os_mark_schema_type	1936

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	operator =	1525

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	IGSetAsListTypedef	1525
ImplOf	1526		

Inherited Protected Data

None

ISetAsTable

ISetAsTable is the table implementation variant of IASet.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isettab.h

ISetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsTable
ISetAsTable

ISetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISetAsTable

Constructs an ISetAsTable object.

Overload 1

```
public:ISetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISetAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

ISetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1940	os_mark_schema_type	1940

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable	1530	operator =	1531

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable	1530	IGSetAsTableTypedef	1531
ImplOf	1531		

Inherited Protected Data

None

ISortedBag

ISortedBag is the default implementation variant, as a list, of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isb.h

ISortedBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedBag
          - IGSortedBag
            - **ISortedBag**

ISortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBag

Constructs an ISortedBag object.

Overload 1

```
public:ISortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBag < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
unionWith	1017		

ISortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1943	os_mark_schema_type	1943

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag	1537	operator =	1537

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag ImplOf	1537 1538	IGSortedBagTypedef	1538

Inherited Protected Data

None

ISortedBagAsAvlTree

ISortedBagAsAvlTree is the AVL tree implementation variant of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isbavl.h

ISortedBagAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedBag
          - IGSortedBagAsAvlTree
            - **ISortedBagAsAvlTree**

ISortedBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBagAsAvlTree

Constructs an ISortedBagAsAvlTree object.

Overload 1

```
public:ISortedBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBagAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

ISortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1947	os_mark_schema_type	1947

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree	1541	operator =	1542

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree	1541	IGSortedBagAsAvlTreeTypedef	1542
ImplOf	1543		

Inherited Protected Data

None

ISortedBagAsBstTree

ISortedBagAsBstTree is the B* tree implementation variant of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isbbst.h

ISortedBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsBstTree
ISortedBagAsBstTree

ISortedBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBagAsBstTree

Constructs an ISortedBagAsBstTree object.

Overload 1

```
public:ISortedBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBagAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

ISortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1951	os_mark_schema_type	1951

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree	1547	operator =	1548

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree ImplOf	1547 1549	IGSortedBagAsBstTreeTypedef	1548

Inherited Protected Data

None

ISortedBagAsDilTable

ISortedBagAsDilTable is the diluted table implementation variant of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isbdil.h

ISortedBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsDilTable
ISortedBagAsDilTable

ISortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBagAsDilTable

Constructs an ISortedBagAsDilTable object.

Overload 1

```
public:ISortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBagAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

ISortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1954	os_mark_schema_type	1954

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable	1554	operator =	1554

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable ImplOf	1554 1555	IGSortedBagAsDilTableTypedef	1554

Inherited Protected Data

None

ISortedBagAsList

ISortedBagAsList is the list implementation variant of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isblst.h

ISortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsList
ISortedBagAsList

ISortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBagAsList

Constructs an ISortedBagAsList object.

Overload 1

```
public:ISortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBagAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
unionWith	1017		

ISortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1958	os_mark_schema_type	1958

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList	1560	operator =	1560

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList ImplOf	1560 1561	IGSortedBagAsListTypedef	1560

Inherited Protected Data

None

ISortedBagAsTable

ISortedBagAsTable is the table implementation variant of IASortedBag.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isbtab.h

ISortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsTable
ISortedBagAsTable

ISortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedBagAsTable

Constructs an ISortedBagAsTable object.

Overload 1

```
public:ISortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedBagAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

ISortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1962	os_mark_schema_type	1962

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable	1566	operator =	1566

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable	1566	IGSortedBagAsTableTypedef	1567
ImplOf	1567		

Inherited Protected Data

None

ISortedMap

ISortedMap is the default implementation variant, as an AVL tree, of IASortedMap.

The implementation of the class ISortedMap requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type

Ordering relation

Class header file: ism.h

ISortedMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMap
ISortedMap

ISortedMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMap

Constructs an ISortedMap object.

Overload 1

```
public:ISortedMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMap < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

ISortedMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

ISortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1966	os_mark_schema_type	1966

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	operator =	1574

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	IGSortedMapTypedef	1574
ImplOf	1575		

Inherited Protected Data

None

ISortedMapAsAvlTree

ISortedMapAsAvlTree is the AVL tree implementation variant of IASortedMap.

The implementation of the class ISortedMapAsAvlTree requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type

Ordering relation

Class header file: ismavl.h

ISortedMapAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsAvlTree
ISortedMapAsAvlTree

ISortedMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMapAsAvlTree

Constructs an ISortedMapAsAvlTree object.

Overload 1

```
public:ISortedMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMapAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

ISortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1970	os_mark_schema_type	1970

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	operator =	1579

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	IGSortedMapAsAvlTreeTypedef	1579
ImplOf	1580		

Inherited Protected Data

None

ISortedMapAsBstTree

ISortedMapAsBstTree is the B* tree implementation variant of IASortedMap.

The implementation of the class ISortedMapAsBstTree requires the following element and key-type functions:

Element Type
- Default constructor
- Copy constructor
- Destructor
- Assignment
- Key access

- Equality test

Key Type

Ordering relation

Class header file: ismbst.h

ISortedMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsBstTree
ISortedMapAsBstTree

ISortedMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMapAsBstTree

Constructs an ISortedMapAsBstTree object.

Overload 1

```
public:ISortedMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMapAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

ISortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1974	os_mark_schema_type	1974

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	operator =	1585

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	IGSortedMapAsBstTreeTypedef	1585
ImplOf	1586		

Inherited Protected Data

None

ISortedMapAsDilTable

ISortedMapAsDilTable is the diluted table implementation variant of IASortedMap.

The implementation of the class ISortedMapAsDilTable requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type

Ordering relation

Class header file: ismdil.h

ISortedMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsDilTable
ISortedMapAsDilTable

ISortedMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMapAsDilTable

Constructs an ISortedMapAsDilTable object.

Overload 1

```
public:ISortedMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMapAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

ISortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1977	os_mark_schema_type	1977

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	operator =	1591

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	IGSortedMapAsDilTableTypedef	1591
ImplOf	1592		

Inherited Protected Data

None

ISortedMapAsList

ISortedMapAsList is the list implementation variant of IASortedMap.

The implementation of the class ISortedMapAsList requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type

Ordering relation

Class header file: ismlst.h

ISortedMapAsList - Hierarchy List

IMetaCollection
  IACollectionBase
    IACollection
      IAEqualityCollection

IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsList
ISortedMapAsList

ISortedMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMapAsList

Constructs an ISortedMapAsList object.

Overload 1

```
public:ISortedMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMapAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

ISortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1981	os_mark_schema_type	1981

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	operator =	1597

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	IGSortedMapAsListTypedef	1597
ImplOf	1598		

Inherited Protected Data

None

ISortedMapAsTable

ISortedMapAsTable is the table implementation variant of IASortedMap.

The implementation of the class ISortedMapAsTable requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type

Ordering relation

Class header file: ismtab.h

ISortedMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsTable
ISortedMapAsTable

ISortedMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedMapAsTable

Constructs an ISortedMapAsTable object.

Overload 1

```
public:ISortedMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedMapAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

ISortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1985	os_mark_schema_type	1985

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsTable	1603	operator =	1603

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	1023 1028 1029 1029	numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	1028 1028 1029

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
`IGSortedMapAsTable` `ImplOf`	1603 1604	`IGSortedMapAsTableTypedef`	1604

Inherited Protected Data

None

ISortedRelation

ISortedRelation is the default implementation variant, as a list, of IASortedRelation.

The default implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type
- Ordering relation

Class header file: isr.h

ISortedRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelation
ISortedRelation

ISortedRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedRelation

Constructs an ISortedRelation object.

Overload 1

```
public:ISortedRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedRelation < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

ISortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1989	os_mark_schema_type	1989

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation	1611	operator =	1611

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation ImplOf	1611 1612	IGSortedRelationTypedef	1612

Inherited Protected Data

None

ISortedRelationAsDilTable

ISortedRelationAsDilTable is the diluted table implementation variant of IASortedRelation.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type
- Ordering relation

Class header file: isrdil.h

ISortedRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsDilTable
ISortedRelationAsDilTable

ISortedRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedRelationAsDilTable

Constructs an ISortedRelationAsDilTable object.

Overload 1

```
public:ISortedRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedRelationAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

ISortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1993	os_mark_schema_type	1993

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable	1615	operator =	1616

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable ImplOf	1615 1617	IGSortedRelationAsDilTableTypedef	1616

Inherited Protected Data

None

ISortedRelationAsTable

ISortedRelationAsTable is the table implementation variant of IASortedRelation.

The implementation of this class requires the following element and key-type functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Key access
- Equality test

Key Type
- Ordering relation

Class header file: isrtab.h

ISortedRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsTable
ISortedRelationAsTable

ISortedRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedRelationAsTable

Constructs an ISortedRelationAsTable object.

Overload 1

```
public:ISortedRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedRelationAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

ISortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1997	os_mark_schema_type	1997

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable	1621	operator =	1622

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable ImplOf	1621 1623	IGSortedRelationAsTableTypedef	1622

Inherited Protected Data

None

ISortedSet

ISortedSet is the default implementation variant, as an AVL tree, of IASortedSet.

The default implementation of the class ISortedSet requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: iss.h

ISortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSet
ISortedSet

ISortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSet

Constructs an ISortedSet object.

Overload 1

```
public:ISortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSet < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

ISortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

ISortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2001	os_mark_schema_type	2001

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet	1629	operator =	1630

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet ImplOf	1629 1630	IGSortedSetTypedef	1629

Inherited Protected Data

None

ISortedSetAsAvlTree

ISortedSetAsAvlTree is the AVL tree implementation variant of IASortedSet.

The implementation of the class ISortedSetAsAvlTree requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: issavl.h

ISortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection

IASortedSet
IGSortedSetAsAvlTree
ISortedSetAsAvlTree

ISortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSetAsAvlTree

Constructs an ISortedSetAsAvlTree object.

Overload 1

```
public:ISortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSetAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet operator != unionWith	1041 1045 1045	intersectionWith operator ==	1045 1045

ISortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2004	os_mark_schema_type	2004

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree	1633	operator =	1634

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree ImplOf	1633 1635	IGSortedSetAsAvlTreeTypedef	1634

Inherited Protected Data

None

ISortedSetAsBstTree

ISortedSetAsBstTree is the B* tree implementation variant of IASortedSet.

The default implementation of the class ISortedSetAsBstTree requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: issbst.h

ISortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsBstTree
ISortedSetAsBstTree

ISortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSetAsBstTree

Constructs an ISortedSetAsBstTree object.

Overload 1

```
public:ISortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSetAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

ISortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2008	os_mark_schema_type	2008

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree	1639	operator =	1640

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree ImplOf	1639 1641	IGSortedSetAsBstTreeTypedef	1640

Inherited Protected Data

None

ISortedSetAsDilTable

ISortedSetAsDilTable is the diluted table implementation variant of IASortedSet.

The implementation of the class ISortedSetAsDilTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: issdil.h

ISortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsDilTable
ISortedSetAsDilTable

ISortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSetAsDilTable

Constructs an ISortedSetAsDilTable object.

Overload 1

```
public:ISortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSetAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

ISortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2012	os_mark_schema_type	2012

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable	1645	operator =	1646

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable ImplOf	1645 1647	IGSortedSetAsDilTableTypedef	1646

Inherited Protected Data

None

ISortedSetAsList

ISortedSetAsList is the list implementation variant of IASortedSet.

The implementation of the class ISortedSetAsList requires the following element functions:

Element Type
- Copy constructor
- Assignment
- Destructor
- Ordering relation

Class header file: isslst.h

ISortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsList
ISortedSetAsList

ISortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSetAsList

Constructs an ISortedSetAsList object.

Overload 1

```
public:ISortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSetAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

ISortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2015	os_mark_schema_type	2015

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList	1652	operator =	1652

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList ImplOf	1652 1653	IGSortedSetAsListTypedef	1652

Inherited Protected Data

None

ISortedSetAsTable

ISortedSetAsTable is the table implementation variant of IASortedSet.

The implementation of the class ISortedSetAsTable requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment
- Ordering relation

Class header file: isstab.h

ISortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsTable
ISortedSetAsTable

ISortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

ISortedSetAsTable

Constructs an ISortedSetAsTable object.

Overload 1

```
public:ISortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:ISortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef ISortedSetAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

ISortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2019	os_mark_schema_type	2019

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable	1658	operator =	1658

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable ImplOf	1658 1659	IGSortedSetAsTableTypedef	1658

Inherited Protected Data

None

IStack

IStack is the default implementation variant, as a list, of IAStack.

The default implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: istk.h

IStack - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStack
IStack

IStack - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IStack

Constructs an IStack object.

Overload 1

```
public:IStack(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IStack(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStack - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IStack < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStack - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IStack			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2023	os_mark_schema_type	2023

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IGStack	1665	operator =	1666

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
`IAStack`	1049		

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
`IGStack` `ImplOf`	1665 1667	`IGStackTypedef`	1666

Inherited Protected Data

None

IStackAsDilTable

IStackAsDilTable is the diluted table implementation variant of IAStack.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: istkdil.h

IStackAsDilTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAStack
        IGStackAsDilTable
          IStackAsDilTable

IStackAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IStackAsDilTable

Constructs an IStackAsDilTable object.

Overload 1

```
public:IStackAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IStackAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IStackAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2026	os_mark_schema_type	2026

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable	1669	operator =	1669

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable ImplOf	1669 1670	IGStackAsDilTableTypedef	1670

Inherited Protected Data

None

IStackAsList

IStackAsList is the list implementation variant of IAStack.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: istklst.h

IStackAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsList
IStackAsList

IStackAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IStackAsList

Constructs an IStackAsList object.

Overload 1

```
public:IStackAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IStackAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IStackAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsList - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2029	os_mark_schema_type	2029

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	operator =	1675

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	IGStackAsListTypedef	1675
ImplOf	1676		

Inherited Protected Data

None

IStackAsTable

IStackAsTable is the table implementation variant of IAStack.

The implementation of this class requires the following element functions:

Element Type
- Copy constructor
- Destructor
- Assignment

Class header file: istktab.h

IStackAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsTable

IStackAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IStackAsTable

Constructs an IStackAsTable object.

Overload 1

```
public:IStackAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IStackAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IStackAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IStackAsTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2032	os_mark_schema_type	2032

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable	1680	operator =	1680

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable	1680	IGStackAsTableTypedef	1680
ImplOf	1681		

Inherited Protected Data

None

IStreamablePointer

A class for wrapping pointers to streamable objects.

Class header file: iptr.h

IStreamablePointer - Hierarchy List

IStreamablePointer

IStreamablePointer - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct streamable pointer objects

~IStreamablePointer

```
public:~IStreamablePointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor.

IStreamablePointer

Constructs an IStreamablePointer object, initializing it with the given pointer, otherwise initializing it to NULL.

Overload 1

```
public:IStreamablePointer(IStreamablePointer < Element > const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IStreamablePointer(Element*, IExplicitInit)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IStreamablePointer()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Operators

Use these operators to access the element being managed.

operator *

```
public:Element& operator *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the object being pointed to.

operator ->

```
public:Element* operator ->() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a raw pointer to the object being pointed to.

operator =

```
public:IStreamablePointer < Element >&
  operator =( IStreamablePointer < Element > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares the pointed to objects for equality.

operator Element *

```
public:operator Element *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts the IStreamablePointer object into a raw pointer.

IStreamablePointer - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITree

A tree is a collection of **nodes** that can have an arbitrary number of **references** to other nodes. There can be no cycles or short-circuit references. A unique path connects every two nodes. One node is designated as the **root** of the tree.

Class header file: itree.h

ITree - Hierarchy List

IMetaTree
  IATreeBase
    IATree
      IAMultiwayTree
        IGMultiwayTree
          IGTree
            ITree

ITree - Member Functions and Data by Group

ITree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946
hasChild	947	numberOfChildren	947

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	1399	IGMultiwayTree	1398
operator =	1398	os_mark_schema_type	1399

IGTree			
Definition	**Page Number**	**Definition**	**Page Number**
operator =	1687		

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
`IATree`	1055	`ImplOf`	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGMultiwayTree` `ImplOf`	1398 1398	`IGMultiwayTreeTypedef`	1398

Inherited Protected Data

None

ITreeCollectionGuard

Use objects of the ITreeCollectionGuard class to ensure the thread safety of a tree collection in a multithreaded program.

The guard object makes sure that only one thread at a time accesses a collection, by locking the collection and preventing other threads from accessing it.

The guard object must be destroyed prior to the destruction of the corresponding collection.

Class header file: iatree.h

ITreeCollectionGuard - Hierarchy List

ITreeCollectionGuard

ITreeCollectionGuard - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

~ITreeCollectionGuard

```
public:~ITreeCollectionGuard()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Unlocks the collection specified within the constructor of this guard.

ITreeCollectionGuard

```
public:ITreeCollectionGuard( IATree < Element >&,
                             ioc::numeric_t timeout = - 1 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The constructor takes the collection object to be locked and an optional timeout value as parameters. The timeout value is specified in milliseconds. If a lock request cannot be resolved within the specified range of time, an exception is thrown. The timeout value defaults to -1. On POSIX systems this parameter is ignored.

ITreeCollectionGuard - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ITreeCursor

For n-ary trees, cursors are used to point to nodes in the tree. Unlike cursors of flat collections, tree cursors stay defined when elements are added to the tree, or when elements othe than the one pointed to are removed. Cursors are used in operations to access the element information stored in a node. They are also used to designate a subtree of the tree, namely the subtree whose root node the cursor points to.

As for flat collections, a distinction is made between the abstract base class ITreeCursor, and cursor classes local to the tree classes themselves. The local, or nested, cursor classes are derived from the abstract base class.

Class header file: ibtree.h

ITreeCursor - Hierarchy List

ITreeCursor

IElementTreeCursor

ITreeCursor - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

~ITreeCursor

```
public:virtual ~ITreeCursor()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this tree.

Side Effects

All cursors of the tree become undefined.

ITreeCursor

Constructs the cursor and associates it with the given tree. The cursor is initially invalid.

Overload 1

```
protected:ITreeCursor(ITreeCursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:ITreeCursor(Implementation*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Cursor Settings

Use these member functions to set the cursor to required position.

setToChild

```
public:bool setToChild(IPosition)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the child with the given position of the node denoted by the given cursor (of this tree). Invalidates the cursor if this child does not exist.

Preconditions

- The cursor must point to an element of this tree.
- (1 <= **position** <= **numberOfChildren**).

Return Value

Returns true if the child exists.

Exceptions

- ICursorInvalidException
- IPositionInvalidException

setToFirstExistingChild

```
public:bool setToFirstExistingChild()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the first child of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no child. A node with no child is a leaf node of the tree.

Preconditions

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a child.

Exception

- ICursorInvalidException

setToLastExistingChild

```
public:bool setToLastExistingChild()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the last child of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no child. A node with no child is a leaf node of the tree.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a child.

Exception
- ICursorInvalidException

setToNextExistingChild

```
public:bool setToNextExistingChild()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the next existing sibling of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no next sibling. A node with no next sibling is the last existing child of its parent.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a next sibling.

Exception
- ICursorInvalidException

setToParent

```
public:bool setToParent()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the parent of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no parent. A node with no parent is the root node of its tree.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a parent.

Exception
- ICursorInvalidException

setToPreviousExistingChild

```
public:bool setToPreviousExistingChild()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the previous existing sibling of the node denoted by the given cursor (of this tree). Invalidates the cursor if the node has no previous sibling. A node with no previous sibling is the first existing child of its parent.

Precondition

The cursor must point to an element of this tree.

Return Value

Returns true if the node has a previous sibling.

Exception
- ICursorInvalidException

setToRoot

```
public:bool setToRoot()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the cursor to the root node of the tree. Invalidates the cursor if the tree is empty (that is, if no root node exists).

Precondition

The cursor must belong to this tree.

Return Value

Returns true if the tree is not empty.

Exception
- ICursorInvalidException

Cursor Validation

Use these member functions to validate the cursor.

invalidate

```
public:void invalidate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Invalidates the cursor so that it no longer points to a node of the associated tree.

isValid

```
public:bool isValid() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to a node of the associated tree.

Miscellaneous

Use these member functions to perform miscellaneous functions.

copy

```
public:void copy(ITreeCursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes all elements from this tree, and copies the subtree denoted by the given cursor of the given tree to (the root of) this tree. The copy function (without a cursor argument) copies the whole given tree.

Preconditions

The cursor must point to an element of the given tree.

Exceptions
- IOutOfMemory
- ICursorInvalidException

ImplOf

```
protected:inline static Implementation& ImplOf(ITreeCursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the implementation object for this tree cursor object.

Operators

Use these members as operators.

operator !=

```
public:bool operator !=(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor does not point to the same node of the same tree as the given cursor.

operator =

```
public:void operator =(ITreeCursor const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copies all elements of the given tree to this tree.

Return Value

A reference to this tree.

Side Effects

All cursors of this tree become undefined.

Exception
- IOutOfMemory

operator ==

```
public:bool operator ==(ITreeCursor const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the cursor points to the same node of the same tree as the given cursor.

ITreeCursor - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IVBag

Class header file: ibag.h

IVBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBag
IVGBag
IVBag

IVBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBag

Constructs an IVBag object.

Overload 1

```
public:IVBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBag < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBag - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2045	os_mark_schema_type	2045

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag	1133	operator =	1134

IVGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBag	2096		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag ImplOf	1133 1135	IGBagTypedef	1134

IVGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBag	2096	IVGBagTypedef	2096

Inherited Protected Data

None

IVBagAsAvlTree

Class header file: ibagavl.h

IVBagAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IABag
        - IGBagAsAvlTree
          - IVGBagAsAvlTree
            - **IVBagAsAvlTree**

IVBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBagAsAvlTree

Constructs an IVBagAsAvlTree object.

Overload 1

```
public:IVBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2048	os_mark_schema_type	2048

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsAvlTree	1137	operator =	1138

IVGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsAvlTree	2099		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsAvlTree ImplOf	1137 1139	IGBagAsAvlTreeTypedef	1138

IVGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsAvlTree	2099	IVGBagAsAvlTreeTypedef	2100

Inherited Protected Data

None

IVBagAsBstTree

Class header file: ibagbst.h

IVBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsBstTree
IVGBagAsBstTree
IVBagAsBstTree

IVBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBagAsBstTree

Constructs an IVBagAsBstTree object.

Overload 1

```
public:IVBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2052	os_mark_schema_type	2052

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsBstTree	1143	operator =	1143

IVGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsBstTree	2103		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsBstTree` `ImplOf`	1143 1144	`IGBagAsBstTreeTypedef`	1144

IVGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGBagAsBstTree`	2103	`IVGBagAsBstTreeTypedef`	2103

Inherited Protected Data

None

IVBagAsDilTable

Variant of IBagAsDilTable enabled for notifications.

Class header file: ibagdil.h

IVBagAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IABag
        - IGBagAsDilTable
          - IVGBagAsDilTable
            - **IVBagAsDilTable**

IVBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IVBagAsDilTable

Constructs an IVBagAsDilTable object.

Overload 1

```
public:IVBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2055	os_mark_schema_type	2055

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable	1148	operator =	1149

IVGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsDilTable	2106		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable ImplOf	1148 1150	IGBagAsDilTableTypedef	1149

IVGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsDilTable	2106	IVGBagAsDilTableTypedef	2107

Inherited Protected Data

None

IVBagAsHshTable

Variant of IBagAsHshTable enabled for notifications.

Class header file: ibaghsh.h

IVBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsHshTable
IVGBagAsHshTable
IVBagAsHshTable

IVBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBagAsHshTable

Constructs an IVBagAsHshTable object.

Overload 1

```
public:IVBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsHshTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2059	os_mark_schema_type	2059

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsHshTable	1154	operator =	1154

IVGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsHshTable	2110		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsHshTable ImplOf	1154 1155	IGBagAsHshTableTypedef	1155

IVGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsHshTable	2110	IVGBagAsHshTableTypedef	2110

Inherited Protected Data

None

IVBagAsList

Variant of IBagAsList enabled for notifications.

Class header file: ibaglst.h

IVBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsList
IVGBagAsList

IVBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBagAsList

Constructs an IVBagAsList object.

Overload 1

```
public:IVBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2062	os_mark_schema_type	2062

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsList	1159	operator =	1160

IVGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsList	2113		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsList ImplOf	1159 1161	IGBagAsListTypedef	1160

IVGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsList	2113	IVGBagAsListTypedef	2114

Inherited Protected Data

None

IVBagAsTable

Variant of IBagAsTable enabled for notifications.

Class header file: ibagtab.h

IVBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsTable
IVGBagAsTable
IVBagAsTable

IVBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVBagAsTable

Constructs an IVBagAsTable object.

Overload 1

```
public:IVBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVBagAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2066	os_mark_schema_type	2066

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable	1165	operator =	1165

IVGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsTable	2117		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable	1165	IGBagAsTableTypedef	1165
ImplOf	1166		

IVGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGBagAsTable	2117	IVGBagAsTableTypedef	2117

Inherited Protected Data

None

IVDeque

Class header file: idqu.h

IVDeque - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDeque
IVGDeque
IVDeque

IVDeque - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVDeque

Constructs an IVDeque object.

Overload 1

```
public:IVDeque(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVDeque(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDeque - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVDeque < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDeque - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDeque - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVDeque			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2069	os_mark_schema_type	2069

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque	1172	operator =	1172

IVGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDeque	2120		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque ImplOf	1172 1173	IGDequeTypedef	1173

IVGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDeque	2120	IVGDequeTypedef	2120

Inherited Protected Data

None

IVDequeAsDilTable

Variant of IDequeAsDilTable enabled for notifications.

Class header file: idqudil.h

IVDequeAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsDilTable
IVGDequeAsDilTable
IVDequeAsDilTable

IVDequeAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVDequeAsDilTable

Constructs an IVDequeAsDilTable object.

Overload 1

```
public:IVDequeAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVDequeAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVDequeAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2073	os_mark_schema_type	2073

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	operator =	1176

IVGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsDilTable	2123		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	IGDequeAsDilTableTypedef	1176
ImplOf	1177		

IVGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsDilTable	2123	IVGDequeAsDilTableTypedef	2124

Inherited Protected Data

None

IVDequeAsList

Variant of IDequeAsList enabled for notifications.

Class header file: idqulst.h

IVDequeAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsList
IVGDequeAsList
IVDequeAsList

IVDequeAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVDequeAsList

Constructs an IVDequeAsList object.

Overload 1

```
public:IVDequeAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVDequeAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVDequeAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsList - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2076	os_mark_schema_type	2076

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList	1181	operator =	1181

IVGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsList	2127		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList ImplOf	1181 1182	IGDequeAsListTypedef	1182

IVGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsList	2127	IVGDequeAsListTypedef	2127

Inherited Protected Data

None

IVDequeAsTable

Variant of IDequeAsTable enabled for notifications.

Class header file: idqutab.h

IVDequeAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsTable
IVGDequeAsTable
IVDequeAsTable

IVDequeAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVDequeAsTable

Constructs an IVDequeAsTable object.

Overload 1

```
public:IVDequeAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVDequeAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVDequeAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVDequeAsTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2079	os_mark_schema_type	2079

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable	1186	operator =	1187

IVGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsTable	2130		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable	1186	IGDequeAsTableTypedef	1187
ImplOf	1188		

IVGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGDequeAsTable	2130	IVGDequeAsTableTypedef	2130

Inherited Protected Data

None

IVEqualitySequence

Variant of IEqualitySequence enabled for notifications.

Class header file: ies.h

IVEqualitySequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence

IGEqualitySequence
IVGEqualitySequence
IVEqualitySequence

IVEqualitySequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVEqualitySequence

Constructs an IVEqualitySequence object.

Overload 1

```
public:IVEqualitySequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVEqualitySequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVEqualitySequence < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member cannot be accessed.

IVEqualitySequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2082	os_mark_schema_type	2082

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequence	1193	operator =	1194

IVGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequence	2133		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequence ImplOf	1193 1195	IGEqualitySequenceTypedef	1194

IVGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequence	2133	IVGEqualitySequenceTypedef	2133

Inherited Protected Data

None

IVEqualitySequenceAsDilTable

Variant of IEqualitySequenceAsDilTable enabled for notifications.

Class header file: iesdil.h

IVEqualitySequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsDilTable
IVGEqualitySequenceAsDilTable
IVEqualitySequenceAsDilTable

IVEqualitySequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVEqualitySequenceAsDilTable

Constructs an IVEqualitySequenceAsDilTable object.

Overload 1

```
public:IVEqualitySequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVEqualitySequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVEqualitySequenceAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2086	os_mark_schema_type	2086

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	operator =	1198

IVGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsDilTable	2136		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	IGEqualitySequenceAsDilTableTypedef	1198
ImplOf	1199		

IVGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsDilTable	2136	IVGEqualitySequenceAsDilTableTypedef	2137

Inherited Protected Data

None

IVEqualitySequenceAsList

Variant of IEqualitySequenceAsList enabled for notifications.

Class header file: ieslst.h

IVEqualitySequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsList
IVGEqualitySequenceAsList
IVEqualitySequenceAsList

IVEqualitySequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVEqualitySequenceAsList

Constructs an IVEqualitySequenceAsList object.

Overload 1

```
public:IVEqualitySequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVEqualitySequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVEqualitySequenceAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
operator !=	898	operator ==	898

IVEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2089	os_mark_schema_type	2089

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList	1203	operator =	1204

IVGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsList	2140		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList ImplOf	1203 1205	IGEqualitySequenceAsListTypedef	1204

IVGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsList	2140	IVGEqualitySequenceAsListTypedef	2140

Inherited Protected Data

None

IVEqualitySequenceAsTable

Variant of IEqualitySequenceAsTable enabled for notifications.

Class header file: iestab.h

IVEqualitySequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsTable
IVGEqualitySequenceAsTable
IVEqualitySequenceAsTable

IVEqualitySequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVEqualitySequenceAsTable

Constructs an IVEqualitySequenceAsTable object.

Overload 1

```
public:IVEqualitySequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVEqualitySequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVEqualitySequenceAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVEqualitySequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2093	os_mark_schema_type	2093

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable	1209	operator =	1209

IVGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsTable	2143		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable ImplOf	1209 1210	IGEqualitySequenceAsTableTypedef	1210

IVGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGEqualitySequenceAsTable	2143	IVGEqualitySequenceAsTableTypedef	2144

Inherited Protected Data

None

IVGBag

Class header file: ibag.h

IVGBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBag
IVGBag
IVBag

IVGBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBag

Constructs an IVGBag object.

Overload 1

```
public:IVGBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGBagTypedef

```
protected:IVGBagTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBag < Element , ElementOps >::PureImplementation
        PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBag - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2097	os_mark_schema_type	2097

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag	1133	operator =	1134

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGBag ImplOf	1133 1135	IGBagTypedef	1134

Inherited Protected Data

None

IVGBagAsAvlTree

Class header file: ibagavl.h

IVGBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsAvlTree
IVGBagAsAvlTree
IVBagAsAvlTree

IVGBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsAvlTree

Constructs an IVGBagAsAvlTree object.

Overload 1

```
public:IVGBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGBagAsAvlTreeTypedef

```
protected:IVGBagAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsAvlTree < Element ,
                        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2100	os_mark_schema_type	2100

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsAvlTree`	1137	`operator =`	1138

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
`~IACollectionBase` `IACollectionBase`	875 875	`CrsrImplOf` `InterfaceFor`	876 876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
`IABag`	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsAvlTree` `ImplOf`	1137 1139	`IGBagAsAvlTreeTypedef`	1138

Inherited Protected Data

None

IVGBagAsBstTree

Class header file: ibagbst.h

IVGBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsBstTree
IVGBagAsBstTree
IVBagAsBstTree

IVGBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsBstTree

Constructs an IVGBagAsBstTree object.

Overload 1

```
public:IVGBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGBagAsBstTreeTypedef

```
protected:IVGBagAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsBstTree < Element ,
                        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2104	os_mark_schema_type	2104

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsBstTree	1143	operator =	1143

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IACollection`	863	`ImplOf`	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsBstTree` `ImplOf`	1143 1144	`IGBagAsBstTreeTypedef`	1144

Inherited Protected Data

None

IVGBagAsDilTable

Class header file: ibagdil.h

IVGBagAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IABag
        - IGBagAsDilTable
          - **IVGBagAsDilTable**
            - IVBagAsDilTable

IVGBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsDilTable

Constructs an IVGBagAsDilTable object.

Overload 1

```
public:IVGBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGBagAsDilTableTypedef

```
protected:IVGBagAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsDilTable < Element ,
                        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2107	os_mark_schema_type	2107

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable	1148	operator =	1149

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsDilTable	1148	IGBagAsDilTableTypedef	1149
ImplOf	1150		

Inherited Protected Data

None

IVGBagAsHshTable

Class header file: ibaghsh.h

IVGBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsHshTable
IVGBagAsHshTable
IVBagAsHshTable

IVGBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsHshTable

Constructs an IVGBagAsHshTable object.

Overload 1

```
public:IVGBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGBagAsHshTableTypedef

```
protected:IVGBagAsHshTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsHshTable < Element ,
                        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2111	os_mark_schema_type	2111

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsHshTable	1154	operator =	1154

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IGBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
`IGBagAsHshTable` `ImplOf`	1154 1155	`IGBagAsHshTableTypedef`	1155

Inherited Protected Data

None

IVGBagAsList

Class header file: ibaglst.h

IVGBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsList
IVGBagAsList
IVBagAsList

IVGBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsList

Constructs an IVGBagAsList object.

Overload 1

```
public:IVGBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGBagAsListTypedef

```
protected:IVGBagAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsList < Element ,
                    ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2114	os_mark_schema_type	2114

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsList	1159	operator =	1160

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsList	1159	IGBagAsListTypedef	1160
ImplOf	1161		

Inherited Protected Data

None

IVGBagAsTable

Class header file: ibagtab.h

IVGBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IABag
IGBagAsTable

IVGBagAsTable
IVBagAsTable

IVGBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGBagAsTable

Constructs an IVGBagAsTable object.

Overload 1

```
public:IVGBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGBagAsTableTypedef

```
protected:IVGBagAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICBagImpl < IVBagImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGBagAsTable < Element , ElementOps >::PureImplementation
        PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
~IABag	855	addDifference	857
addIntersection	857	addUnion	858
differenceWith	859	IABag	856
intersectionWith	859	numberOfDifferentElements	859
operator !=	859	operator ==	859
setToNextDifferentElement	860	unionWith	860

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IVGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2117	os_mark_schema_type	2118

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable	1165	operator =	1165

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IABag			
Definition	**Page Number**	**Definition**	**Page Number**
IABag	856		

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IGBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGBagAsTable ImplOf	1165 1166	IGBagAsTableTypedef	1165

Inherited Protected Data

None

IVGDeque

Class header file: idqu.h

IVGDeque - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDeque
IVGDeque
IVDeque

IVGDeque - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGDeque

Constructs an IVGDeque object.

Overload 1

```
public:IVGDeque(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGDeque(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGDequeTypedef

```
protected:IVGDequeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDeque - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGDeque < Element , ElementOps >::PureImplementation
        PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDeque - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDeque - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2121	os_mark_schema_type	2121

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque	1172	operator =	1172

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDeque			
Definition	**Page Number**	**Definition**	**Page Number**
IGDeque ImplOf	1172 1173	IGDequeTypedef	1173

Inherited Protected Data

None

IVGDequeAsDilTable

Class header file: idqudil.h

IVGDequeAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsDilTable
IVGDequeAsDilTable
IVDequeAsDilTable

IVGDequeAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGDequeAsDilTable

Constructs an IVGDequeAsDilTable object.

Overload 1

```
public:IVGDequeAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGDequeAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGDequeAsDilTableTypedef

```
protected:IVGDequeAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGDequeAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2124	os_mark_schema_type	2124

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	operator =	1176

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsDilTable	1175	IGDequeAsDilTableTypedef	1176
ImplOf	1177		

Inherited Protected Data

None

IVGDequeAsList

Class header file: idqulst.h

IVGDequeAsList - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IADeque
        IGDequeAsList
          IVGDequeAsList
            IVDequeAsList

IVGDequeAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGDequeAsList

Constructs an IVGDequeAsList object.

Overload 1

```
public:IVGDequeAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGDequeAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGDequeAsListTypedef

```
protected:IVGDequeAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGDequeAsList < Element ,
             ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsList - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2127	os_mark_schema_type	2127

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList	1181	operator =	1181

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsList	1181	IGDequeAsListTypedef	1182
ImplOf	1182		

Inherited Protected Data

None

IVGDequeAsTable

Class header file: idqutab.h

IVGDequeAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IADeque
IGDequeAsTable
IVGDequeAsTable
IVDequeAsTable

IVGDequeAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGDequeAsTable

Constructs an IVGDequeAsTable object.

Overload 1

```
public:IVGDequeAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGDequeAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGDequeAsTableTypedef

```
protected:IVGDequeAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGDequeAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGDequeAsTable - Inherited Member Functions and Data

Inherited Public Functions

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
~IADeque	878	addAsFirst	879
addAsLast	879	compare	880
IADeque	878	removeFirst	881
removeLast	881		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2131	os_mark_schema_type	2131

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable	1186	operator =	1187

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IADeque			
Definition	**Page Number**	**Definition**	**Page Number**
IADeque	878		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGDequeAsTable	1186	IGDequeAsTableTypedef	1187

IGDequeAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1188		

Inherited Protected Data

None

IVGEqualitySequence

Class header file: ies.h

IVGEqualitySequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequence
IVGEqualitySequence
IVEqualitySequence

IVGEqualitySequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGEqualitySequence

Constructs an IVGEqualitySequence object.

Overload 1

```
public:IVGEqualitySequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGEqualitySequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef ICEqualitySequenceImpl < IVEqualitySequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGEqualitySequence < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2134	os_mark_schema_type	2134

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequence	1193	operator =	1194

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualityCollection`	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
`IAEqualitySequence`	895		

IGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
`IGEqualitySequence` `ImplOf`	1193 1195	`IGEqualitySequenceTypedef`	1194

IVGEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGEqualitySequenceTypedef`	2133		

Inherited Protected Data

None

IVGEqualitySequenceAsDilTable

Class header file: iesdil.h

IVGEqualitySequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsDilTable
IVGEqualitySequenceAsDilTable
IVEqualitySequenceAsDilTable

IVGEqualitySequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IVGEqualitySequenceAsDilTable

Constructs an IVGEqualitySequenceAsDilTable object.

Overload 1

```
public:IVGEqualitySequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGEqualitySequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGEqualitySequenceAsDilTableTypedef

```
protected:IVGEqualitySequenceAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICEqualitySequenceImpl < IVEqualitySequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGEqualitySequenceAsDilTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2137	os_mark_schema_type	2137

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	operator =	1198

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsDilTable	1197	IGEqualitySequenceAsDilTableTypedef	1198
ImplOf	1199		

Inherited Protected Data

None

IVGEqualitySequenceAsList

Class header file: ieslst.h

IVGEqualitySequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsList
IVGEqualitySequenceAsList
IVEqualitySequenceAsList

IVGEqualitySequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGEqualitySequenceAsList

Constructs an IVGEqualitySequenceAsList object.

Overload 1

```
public:IVGEqualitySequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGEqualitySequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGEqualitySequenceAsListTypedef

```
protected:IVGEqualitySequenceAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICEqualitySequenceImpl < IVEqualitySequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGEqualitySequenceAsList < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2141	os_mark_schema_type	2141

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList	1203	operator =	1204

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsList ImplOf	1203 1205	IGEqualitySequenceAsListTypedef	1204

Inherited Protected Data

None

IVGEqualitySequenceAsTable

Class header file: iestab.h

IVGEqualitySequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySequence
IGEqualitySequenceAsTable
IVGEqualitySequenceAsTable
IVEqualitySequenceAsTable

IVGEqualitySequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGEqualitySequenceAsTable

Constructs an IVGEqualitySequenceAsTable object.

Overload 1

```
public:IVGEqualitySequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGEqualitySequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGEqualitySequenceAsTableTypedef

```
protected:IVGEqualitySequenceAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICEqualitySequenceImpl < IVEqualitySequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVEqualitySequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGEqualitySequenceAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGEqualitySequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySequence	895	compare	896
IAEqualitySequence	895	locateFirst	897
locateLast	897	locatePrevious	897
operator !=	898	operator ==	898

IVGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2144	os_mark_schema_type	2144

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable	1209	operator =	1209

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySequence			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySequence	895		

IGEqualitySequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGEqualitySequenceAsTable ImplOf	1209 1210	IGEqualitySequenceAsTableTypedef	1210

Inherited Protected Data

None

IVGHeap

Class header file: ihp.h

IVGHeap - Hierarchy List

IGHeap

IVGHeap

IVHeap

IVGHeap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGHeap

Constructs an IVGHeap object.

Overload 1

```
public:IVGHeap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGHeap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGHeapTypedef

```
protected:IVGHeapTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICHeapImpl < IVHeapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGHeap < Element , ElementOps >::PureImplementation
        PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeap - Inherited Member Functions and Data

Inherited Public Functions

IVGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2148	os_mark_schema_type	2148

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap	1216	operator =	1217

Inherited Public Data

None

Inherited Protected Functions

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap ImplOf	1216 1218	IGHeapTypedef	1217

Inherited Protected Data

None

IVGHeapAsDilTable

Class header file: ihpdil.h

IVGHeapAsDilTable - Hierarchy List

IGHeapAsDilTable

IVGHeapAsDilTable

IVGHeapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGHeapAsDilTable

Constructs an IVGHeapAsDilTable object.

Overload 1

```
public:IVGHeapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGHeapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGHeapAsDilTableTypedef

```
protected:IVGHeapAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICHeapImpl < IVHeapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGHeapAsDilTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IVGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2149	os_mark_schema_type	2150

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable	1219	operator =	1219

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable ImplOf	1219 1220	IGHeapAsDilTableTypedef	1219

Inherited Protected Data

None

IVGHeapAsList

Class header file: ihplst.h

IVGHeapAsList - Hierarchy List

IGHeapAsList

IVGHeapAsList

IVHeapAsList

IVGHeapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGHeapAsList

Constructs an IVGHeapAsList object.

Overload 1

```
public:IVGHeapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGHeapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGHeapAsListTypedef

```
protected:IVGHeapAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICHeapImpl < IVHeapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGHeapAsList < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsList - Inherited Member Functions and Data

Inherited Public Functions

IVGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2151	os_mark_schema_type	2151

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList	1222	operator =	1223

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList ImplOf	1222 1224	IGHeapAsListTypedef	1223

Inherited Protected Data

None

IVGHeapAsTable

Class header file: ihptab.h

IVGHeapAsTable - Hierarchy List

IGHeapAsTable

IVGHeapAsTable

IVHeapAsTable

IVGHeapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGHeapAsTable

Constructs an IVGHeapAsTable object.

Overload 1

```
public:IVGHeapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGHeapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGHeapAsTableTypedef

```
protected:IVGHeapAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICHeapImpl < IVHeapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVHeapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGHeapAsTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGHeapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IVGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2153	os_mark_schema_type	2153

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsTable	1226	operator =	1227

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsTable ImplOf	1226 1227	IGHeapAsTableTypedef	1227

Inherited Protected Data

None

IVGKeyBag

Class header file: ikb.h

IVGKeyBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBag
IVGKeyBag
IVKeyBag

IVGKeyBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeyBag

Constructs an IVGKeyBag object.

Overload 1

```
public:IVGKeyBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeyBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeyBagTypedef

```
protected:IVGKeyBagTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < IVKeyBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeyBag < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2155	os_mark_schema_type	2155

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBag	1231	operator =	1232

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBag	1231	IGKeyBagTypedef	1232
ImplOf	1233		

Inherited Protected Data

None

IVGKeyBagAsDilTable

Class header file: ikbdil.h

IVGKeyBagAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeyBag
        - IGKeyBagAsDilTable

IVGKeyBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeyBagAsDilTable

Constructs an IVGKeyBagAsDilTable object.

Overload 1

```
public:IVGKeyBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeyBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeyBagAsDilTableTypedef

```
protected:IVGKeyBagAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < IVKeyBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeyBagAsDilTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2158	os_mark_schema_type	2159

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable	1235	operator =	1236

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable	1235	IGKeyBagAsDilTableTypedef	1236
ImplOf	1236		

Inherited Protected Data

None

IVGKeyBagAsHshTable

Class header file: ikbhsh.h

IVGKeyBagAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsHshTable
IVGKeyBagAsHshTable
IVKeyBagAsHshTable

IVGKeyBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeyBagAsHshTable

Constructs an IVGKeyBagAsHshTable object.

Overload 1

```
public:IVGKeyBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeyBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeyBagAsHshTableTypedef

```
protected:IVGKeyBagAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeyBagImpl < IVKeyBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeyBagAsHshTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2162	os_mark_schema_type	2162

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable	1240	operator =	1241

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable ImplOf	1240 1242	IGKeyBagAsHshTableTypedef	1241

Inherited Protected Data

None

IVGKeyBagAsList

Class header file: ikblst.h

IVGKeyBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsList
IVGKeyBagAsList
IVKeyBagAsList

IVGKeyBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeyBagAsList

Constructs an IVGKeyBagAsList object.

Overload 1

```
public:IVGKeyBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeyBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeyBagAsListTypedef

```
protected:IVGKeyBagAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeyBagImpl < IVKeyBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeyBagAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2165	os_mark_schema_type	2165

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList	1246	operator =	1246

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList ImplOf	1246 1247	IGKeyBagAsListTypedef	1246

Inherited Protected Data

None

IVGKeyBagAsTable

Class header file: ikbtab.h

IVGKeyBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsTable
IVGKeyBagAsTable
IVKeyBagAsTable

IVGKeyBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeyBagAsTable

Constructs an IVGKeyBagAsTable object.

Overload 1

```
public:IVGKeyBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeyBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeyBagAsTableTypedef

```
protected:IVGKeyBagAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeyBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeyBagImpl < IVKeyBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeyBagAsTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeyBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2169	os_mark_schema_type	2169

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable	1251	operator =	1252

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable ImplOf	1251 1253	IGKeyBagAsTableTypedef	1252

Inherited Protected Data

None

IVGKeySet

Variant of IKeySet enabled for notifications.

Class header file: iks.h

IVGKeySet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySet
IVGKeySet
IVKeySet

IVGKeySet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySet

Constructs an IVGKeySet object.

Overload 1

```
public:IVGKeySet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySet < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2172	os_mark_schema_type	2172

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	operator =	1259

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	IGKeySetTypedef	1258
ImplOf	1260		

IVGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetTypedef	2171		

Inherited Protected Data

None

IVGKeySetAsAvlTree

Class header file: iksavl.h

IVGKeySetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsAvlTree
IVGKeySetAsAvlTree
IVKeySetAsAvlTree

IVGKeySetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsAvlTree

Constructs an IVGKeySetAsAvlTree object.

Overload 1

```
public:IVGKeySetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsAvlTreeTypedef

```
protected:IVGKeySetAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsAvlTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2176	os_mark_schema_type	2176

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree	1262	operator =	1263

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree	1262	IGKeySetAsAvlTreeTypedef	1263
ImplOf	1264		

Inherited Protected Data

None

IVGKeySetAsBstTree

Class header file: iksbst.h

IVGKeySetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsBstTree
IVGKeySetAsBstTree
IVKeySetAsBstTree

IVGKeySetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsBstTree

Constructs an IVGKeySetAsBstTree object.

Overload 1

```
public:IVGKeySetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsBstTreeTypedef

```
protected:IVGKeySetAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsBstTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2179	os_mark_schema_type	2179

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree	1268	operator =	1268

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree	1268	IGKeySetAsBstTreeTypedef	1268
ImplOf	1269		

Inherited Protected Data

None

IVGKeySetAsDilTable

Class header file: iksdil.h

IVGKeySetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsDilTable
IVGKeySetAsDilTable
IVKeySetAsDilTable

IVGKeySetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsDilTable

Constructs an IVGKeySetAsDilTable object.

Overload 1

```
public:IVGKeySetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsDilTableTypedef

```
protected:IVGKeySetAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsDilTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2183	os_mark_schema_type	2183

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	operator =	1274

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	IGKeySetAsDilTableTypedef	1274
ImplOf	1275		

Inherited Protected Data

None

IVGKeySetAsHshTable

Class header file: ikshsh.h

IVGKeySetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsHshTable
IVGKeySetAsHshTable
IVKeySetAsHshTable

IVGKeySetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsHshTable

Constructs an IVGKeySetAsHshTable object.

Overload 1

```
public:IVGKeySetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsHshTableTypedef

```
protected:IVGKeySetAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsHshTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2186	os_mark_schema_type	2186

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable	1279	operator =	1279

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable	1279	IGKeySetAsHshTableTypedef	1279
ImplOf	1280		

Inherited Protected Data

None

IVGKeySetAsList

Class header file: ikslst.h

IVGKeySetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsList
IVGKeySetAsList
IVKeySetAsList

IVGKeySetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsList

Constructs an IVGKeySetAsList object.

Overload 1

```
public:IVGKeySetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsListTypedef

```
protected:IVGKeySetAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2190	os_mark_schema_type	2190

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	operator =	1285

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	IGKeySetAsListTypedef	1285
ImplOf	1286		

Inherited Protected Data

None

IVGKeySetAsTable

Class header file: ikstab.h

IVGKeySetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsTable
IVGKeySetAsTable
IVKeySetAsTable

IVGKeySetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySetAsTable

Constructs an IVGKeySetAsTable object.

Overload 1

```
public:IVGKeySetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySetAsTableTypedef

```
protected:IVGKeySetAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySetImpl < IVKeySetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySetAsTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2193	os_mark_schema_type	2193

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable	1290	operator =	1290

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable	1290	IGKeySetAsTableTypedef	1290
ImplOf	1291		

Inherited Protected Data

None

IVGKeySortedBag

Variant of IKeySortedBag enabled for notifications.

Class header file: iksb.h

IVGKeySortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBag
IVGKeySortedBag
IVKeySortedBag

IVGKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedBag

Constructs an IVGKeySortedBag object.

Overload 1

```
public:IVGKeySortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedBag < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2197	os_mark_schema_type	2197

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag	1297	operator =	1297

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag ImplOf	1297 1298	IGKeySortedBagTypedef	1296

IVGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagTypedef	2196		

Inherited Protected Data

None

IVGKeySortedBagAsDilTable

Class header file: iksbdil.h

IVGKeySortedBagAsDilTable - Hierarchy List

IMetaCollection

IACollectionBase

IACollection

IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsDilTable
IVGKeySortedBagAsDilTable
IVKeySortedBagAsDilTable

IVGKeySortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedBagAsDilTable

Constructs an IVGKeySortedBagAsDilTable object.

Overload 1

```
public:IVGKeySortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySortedBagAsDilTableTypedef

```
protected:IVGKeySortedBagAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedBagAsDilTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag IAKeySortedBag	921 922	compare	923

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2201	os_mark_schema_type	2201

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable	1301	operator =	1302

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable ImplOf	1301 1303	IGKeySortedBagAsDilTableTypedef	1302

Inherited Protected Data

None

IVGKeySortedBagAsList

Class header file: iksblst.h

IVGKeySortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsList
IVGKeySortedBagAsList
IVKeySortedBagAsList

IVGKeySortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedBagAsList

Constructs an IVGKeySortedBagAsList object.

Overload 1

```
public:IVGKeySortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySortedBagAsListTypedef

```
protected:IVGKeySortedBagAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedBagAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag IAKeySortedBag	921 922	compare	923

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2204	os_mark_schema_type	2204

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList	1307	operator =	1308

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList ImplOf	1307 1309	IGKeySortedBagAsListTypedef	1308

Inherited Protected Data

None

IVGKeySortedBagAsTable

Class header file: iksbtab.h

IVGKeySortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsTable
IVGKeySortedBagAsTable
IVKeySortedBagAsTable

IVGKeySortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedBagAsTable

Constructs an IVGKeySortedBagAsTable object.

Overload 1

```
public:IVGKeySortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGKeySortedBagAsTableTypedef

```
protected:IVGKeySortedBagAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedBagAsTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2208	os_mark_schema_type	2208

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable	1313	operator =	1314

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable ImplOf	1313 1315	IGKeySortedBagAsTableTypedef	1314

Inherited Protected Data

None

IVGKeySortedSet

Class header file: ikss.h

IVGKeySortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSet
IVGKeySortedSet
IVKeySortedSet

IVGKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSet

Constructs an IVGKeySortedSet object.

Overload 1

```
public:IVGKeySortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetTypedef

```
protected:IVGKeySortedSetTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSet < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2212	os_mark_schema_type	2212

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet	1321	operator =	1322

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet ImplOf	1321 1323	IGKeySortedSetTypedef	1322

Inherited Protected Data

None

IVGKeySortedSetAsAvlTree

Class header file: ikssavl.h

IVGKeySortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsAvlTree
IVGKeySortedSetAsAvlTree
IVKeySortedSetAsAvlTree

IVGKeySortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSetAsAvlTree

Constructs an IVGKeySortedSetAsAvlTree object.

Overload 1

```
public:IVGKeySortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetAsAvlTreeTypedef

```
protected:IVGKeySortedSetAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSetAsAvlTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2216	os_mark_schema_type	2216

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree	1326	operator =	1326

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree	1326	IGKeySortedSetAsAvlTreeTypedef	1326
ImplOf	1327		

Inherited Protected Data

None

IVGKeySortedSetAsBstTree

Class header file: ikssbst.h

IVGKeySortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsBstTree
IVGKeySortedSetAsBstTree
IVKeySortedSetAsBstTree

IVGKeySortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSetAsBstTree

Constructs an IVGKeySortedSetAsBstTree object.

Overload 1

```
public:IVGKeySortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetAsBstTreeTypedef

```
protected:IVGKeySortedSetAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSetAsBstTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2220	os_mark_schema_type	2220

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree	1332	operator =	1332

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree	1332	IGKeySortedSetAsBstTreeTypedef	1332
ImplOf	1333		

Inherited Protected Data

None

IVGKeySortedSetAsDilTable

Class header file: ikssdil.h

IVGKeySortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsDilTable

IVGKeySortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSetAsDilTable

Constructs an IVGKeySortedSetAsDilTable object.

Overload 1

```
public:IVGKeySortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetAsDilTableTypedef

```
protected:IVGKeySortedSetAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSetAsDilTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IVGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2224	os_mark_schema_type	2224

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable	1338	operator =	1338

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable ImplOf	1338 1339	IGKeySortedSetAsDilTableTypedef	1338

Inherited Protected Data

None

IVGKeySortedSetAsList

Class header file: iksslst.h

IVGKeySortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsList
IVGKeySortedSetAsList
IVKeySortedSetAsList

IVGKeySortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSetAsList

Constructs an IVGKeySortedSetAsList object.

Overload 1

```
public:IVGKeySortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetAsListTypedef

```
protected:IVGKeySortedSetAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSetAsList < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IVGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2227	os_mark_schema_type	2227

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList	1344	operator =	1344

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList ImplOf	1344 1345	IGKeySortedSetAsListTypedef	1344

Inherited Protected Data

None

IVGKeySortedSetAsTable

Class header file: iksstab.h

IVGKeySortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsTable
IVGKeySortedSetAsTable
IVKeySortedSetAsTable

IVGKeySortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGKeySortedSetAsTable

Constructs an IVGKeySortedSetAsTable object.

Overload 1

```
public:IVGKeySortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGKeySortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGKeySortedSetAsTableTypedef

```
protected:IVGKeySortedSetAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedSetImpl < IVKeySortedSetImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGKeySortedSetAsTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGKeySortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IVGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2231	os_mark_schema_type	2231

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable	1350	operator =	1350

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable	1350	IGKeySortedSetAsTableTypedef	1350
ImplOf	1351		

Inherited Protected Data

None

IVGMap

Class header file: imap.h

IVGMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMap
IVGMap
IVMap

IVGMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMap

Constructs an IVGMap object.

Overload 1

```
public:IVGMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapTypedef

```
protected:IVGMapTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < IVMapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMap < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2235	os_mark_schema_type	2235

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGMap	1357	operator =	1358

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap locateNextElementWithKey numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	934 939 940 940 941	locateNext numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	939 940 940 941

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
`IGMap` `ImplOf`	1357 1359	`IGMapTypedef`	1358

Inherited Protected Data

None

IVGMapAsAvlTree

Class header file: imapavl.h

IVGMapAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsAvlTree
IVGMapAsAvlTree
IVMapAsAvlTree

IVGMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsAvlTree

Constructs an IVGMapAsAvlTree object.

Overload 1

```
public:IVGMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsAvlTreeTypedef

```
protected:IVGMapAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < IVMapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsAvlTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2239	os_mark_schema_type	2239

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree	1362	operator =	1362

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree	1362	IGMapAsAvlTreeTypedef	1362
ImplOf	1363		

Inherited Protected Data

None

IVGMapAsBstTree

Class header file: imapbst.h

IVGMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsBstTree
IVGMapAsBstTree
IVMapAsBstTree

IVGMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsBstTree

Constructs an IVGMapAsBstTree object.

Overload 1

```
public:IVGMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsBstTreeTypedef

```
protected:IVGMapAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < IVMapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsBstTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2243	os_mark_schema_type	2243

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsBstTree	1367	operator =	1368

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsBstTree	1367	IGMapAsBstTreeTypedef	1368
ImplOf	1369		

Inherited Protected Data

None

IVGMapAsDilTable

Class header file: imapdil.h

IVGMapAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection

IAMap
IGMapAsDilTable
IVGMapAsDilTable
IVMapAsDilTable

IVGMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsDilTable

Constructs an IGMapAsDilTable object.

Overload 1

```
public:IVGMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsDilTableTypedef

```
protected:IVGMapAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < IVMapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsDilTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2247	os_mark_schema_type	2247

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsDilTable	1373	operator =	1374

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsDilTable	1373	IGMapAsDilTableTypedef	1374
ImplOf	1374		

Inherited Protected Data

None

IVGMapAsHshTable

Class header file: imaphsh.h

IVGMapAsHshTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IAMap
          - IGMapAsHshTable
            - **IVGMapAsHshTable**
              - IVMapAsHshTable

IVGMapAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsHshTable

Constructs an IVGMapAsHshTable object.

Overload 1

```
public:IVGMapAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsHshTableTypedef

```
protected:IVGMapAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < IVMapImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsHshTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	938	unionWith	938

IVGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2250	os_mark_schema_type	2250

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable	1379	operator =	1379

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap locateNextElementWithKey numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	934 939 940 940 941	locateNext numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	939 940 940 941

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable ImplOf	1379 1380	IGMapAsHshTableTypedef	1379

Inherited Protected Data

None

IVGMapAsList

Class header file: imaplst.h

IVGMapAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsList
IVGMapAsList
IVMapAsList

IVGMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsList

Constructs an IVGMapAsList object.

Overload 1

```
public:IVGMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsListTypedef

```
protected:IVGMapAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICMapImpl < IVMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2254	os_mark_schema_type	2254

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList	1385	operator =	1385

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	941		

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList ImplOf	1385 1386	IGMapAsListTypedef	1385

Inherited Protected Data

None

IVGMapAsTable

Class header file: imaptab.h

IVGMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsTable
IVGMapAsTable
IVMapAsTable

IVGMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGMapAsTable

Constructs an IVGMapAsTable object.

Overload 1

```
public:IVGMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGMapAsTableTypedef

```
protected:IVGMapAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICMapImpl < IVMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMapAsTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2258	os_mark_schema_type	2258

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable	1391	operator =	1391

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable ImplOf	1391 1392	IGMapAsTableTypedef	1391

Inherited Protected Data

None

IVGMultiwayTree

Notification-enabled variant of IMultiwayTree.

Class header file: imwt.h

IVGMultiwayTree - Hierarchy List

IMetaTree
IATreeBase
IATree
IAMultiwayTree
IGMultiwayTree
IVGMultiwayTree
IVMultiwayTree

IVGMultiwayTree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IVGMultiwayTree

Constructs an IVGMultiwayTree object.

Overload 1

```
public:IVGMultiwayTree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IVGMultiwayTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMultiwayTree - Type Definitions

Guard

```
typedef ITreeCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef ICMultiwayTreeImpl < IVMultiwayTreeImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVMultiwayTreeImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGMultiwayTree < numOfChildren , Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMultiwayTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGMultiwayTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946
hasChild	947	numberOfChildren	947

IVGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2262	os_mark_schema_type	2262

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree	1398	operator =	1398

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree	1398	IGMultiwayTreeTypedef	1398

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
`ImplOf`	1398		

IVGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGMultiwayTreeTypedef`	2261		

Inherited Protected Data

None

IVGPriorityQueue

Class header file: ipqu.h

IVGPriorityQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueue
IVGPriorityQueue
IVPriorityQueue

IVGPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGPriorityQueue

Constructs an IVGPriorityQueue object.

Overload 1

```
public:IVGPriorityQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGPriorityQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGPriorityQueueTypedef

```
protected:IVGPriorityQueueTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGPriorityQueue < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2265	os_mark_schema_type	2265

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue	1403	operator =	1403

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue	1403	IGPriorityQueueTypedef	1403
ImplOf	1404		

Inherited Protected Data

None

IVGPriorityQueueAsDilTable

Class header file: ipqudil.h

IVGPriorityQueueAsDilTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAPriorityQueue
        IGPriorityQueueAsDilTable
          IVGPriorityQueueAsDilTable
            IVPriorityQueueAsDilTable

IVGPriorityQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGPriorityQueueAsDilTable

Constructs an IVGPriorityQueueAsDilTable object.

Overload 1

```
public:IVGPriorityQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGPriorityQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGPriorityQueueAsDilTableTypedef

```
protected:IVGPriorityQueueAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGPriorityQueueAsDilTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext setToPrevious	984 985	setToPosition	985

IVGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2268	os_mark_schema_type	2269

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable	1407	operator =	1407

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable ImplOf	1407 1408	IGPriorityQueueAsDilTableTypedef	1407

Inherited Protected Data

None

IVGPriorityQueueAsList

Class header file: ipqulst.h

IVGPriorityQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsList
IVGPriorityQueueAsList
IVPriorityQueueAsList

IVGPriorityQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGPriorityQueueAsList

Constructs an IVGPriorityQueueAsList object.

Overload 1

```
public:IVGPriorityQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGPriorityQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGPriorityQueueAsListTypedef

```
protected:IVGPriorityQueueAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGPriorityQueueAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2272	os_mark_schema_type	2272

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList	1412	operator =	1413

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList ImplOf	1412 1414	IGPriorityQueueAsListTypedef	1413

Inherited Protected Data

None

IVGPriorityQueueAsTable

Class header file: ipqutab.h

IVGPriorityQueueAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsTable
IVGPriorityQueueAsTable
IVPriorityQueueAsTable

IVGPriorityQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGPriorityQueueAsTable

Constructs an IVGPriorityQueueAsTable object.

Overload 1

```
public:IVGPriorityQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGPriorityQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsTableTypedef

```
protected:IVGPriorityQueueAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVKeySortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICKeySortedBagImpl < IVKeySortedBagImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGPriorityQueueAsTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGPriorityQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2275	os_mark_schema_type	2275

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable	1418	operator =	1418

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable	1418	IGPriorityQueueAsTableTypedef	1419
ImplOf	1419		

Inherited Protected Data

None

IVGQueue

Class header file: iqu.h

IVGQueue - Hierarchy List

IMetaCollection
- IACollectionBase
  - IARestrictedAccessCollection
    - IAQueue
      - IGQueue

IVGQueue

IVQueue

IVGQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGQueue

Constructs an IVGQueue object.

Overload 1

```
public:IVGQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueTypedef

```
protected:IVGQueueTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGQueue < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueue - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2278	os_mark_schema_type	2279

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue	1425	operator =	1426

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue	1425	IGQueueTypedef	1426
ImplOf	1426		

Inherited Protected Data

None

IVGQueueAsDilTable

Class header file: iqudil.h

IVGQueueAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsDilTable
IVGQueueAsDilTable
IVQueueAsDilTable

IVGQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGQueueAsDilTable

Constructs an IVGQueueAsDilTable object.

Overload 1

```
public:IVGQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGQueueAsDilTableTypedef

```
protected:IVGQueueAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < IVSequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGQueueAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2282	os_mark_schema_type	2282

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable	1429	operator =	1429

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable ImplOf	1429 1430	IGQueueAsDilTableTypedef	1429

Inherited Protected Data

None

IVGQueueAsList

Class header file: iqulst.h

IVGQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsList
IVGQueueAsList
IVQueueAsList

IVGQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGQueueAsList

Constructs an IVGQueueAsList object.

Overload 1

```
public:IVGQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGQueueAsListTypedef

```
protected:IVGQueueAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGQueueAsList < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2285	os_mark_schema_type	2285

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList	1434	operator =	1435

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList ImplOf	1434 1436	IGQueueAsListTypedef	1435

Inherited Protected Data

None

IVGQueueAsTable

Class header file: iqutab.h

IVGQueueAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsTable
IVGQueueAsTable
IVQueueAsTable

IVGQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGQueueAsTable

Constructs an IVGQueueAsTable object.

Overload 1

```
public:IVGQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGQueueAsTableTypedef

```
protected:IVGQueueAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGQueueAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2288	os_mark_schema_type	2288

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	operator =	1440

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	IGQueueAsTableTypedef	1440
ImplOf	1441		

Inherited Protected Data

None

IVGRelation

Class header file: irel.h

IVGRelation - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IARelation
          - IGRelation
            - **IVGRelation**
              - IVRelation

IVGRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGRelation

Constructs an IVGRelation object.

Overload 1

```
public:IVGRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGRelationTypedef

```
protected:IVGRelationTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICRelationImpl < IVRelationImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGRelation < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2291	os_mark_schema_type	2291

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation	1447	operator =	1447

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation ImplOf	1447 1448	IGRelationTypedef	1448

Inherited Protected Data

None

IVGRelationAsDilTable

Class header file: ireldil.h

IVGRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsDilTable
IVGRelationAsDilTable
IVRelationAsDilTable

IVGRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGRelationAsDilTable

Constructs an IVGRelationAsDilTable object.

Overload 1

```
public:IVGRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGRelationAsDilTableTypedef

```
protected:IVGRelationAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICRelationImpl < IVRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGRelationAsDilTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2295	os_mark_schema_type	2295

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable	1451	operator =	1452

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable ImplOf	1451 1452	IGRelationAsDilTableTypedef	1452

Inherited Protected Data

None

IVGRelationAsHshTable

Class header file: irelhsh.h

IVGRelationAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsHshTable
IVGRelationAsHshTable
IVRelationAsHshTable

IVGRelationAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGRelationAsHshTable

Constructs an IVGRelationAsHshTable object.

Overload 1

```
public:IVGRelationAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGRelationAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGRelationAsHshTableTypedef

```
protected:IVGRelationAsHshTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < IVRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGRelationAsHshTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2299	os_mark_schema_type	2299

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable	1457	operator =	1457

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable	1457	IGRelationAsHshTableTypedef	1457
ImplOf	1458		

Inherited Protected Data

None

IVGRelationAsList

Class header file: irellst.h

IVGRelationAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsList
IVGRelationAsList
IVRelationAsList

IVGRelationAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGRelationAsList

Constructs an IVGRelationAsList object.

Overload 1

```
public:IVGRelationAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGRelationAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Miscellaneous Members

IVGRelationAsListTypedef

```
protected:IVGRelationAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICRelationImpl < IVRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGRelationAsList < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2303	os_mark_schema_type	2303

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList	1462	operator =	1463

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList	1462	IGRelationAsListTypedef	1463
ImplOf	1464		

Inherited Protected Data

None

IVGRelationAsTable

Class header file: ireltab.h

IVGRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsTable
IVGRelationAsTable
IVRelationAsTable

IVGRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGRelationAsTable

Constructs an IVGRelationAsTable object.

Overload 1

```
public:IVGRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGRelationAsTableTypedef

```
protected:IVGRelationAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICRelationImpl < IVRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGRelationAsTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2306	os_mark_schema_type	2306

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable	1468	operator =	1468

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable ImplOf	1468 1469	IGRelationAsTableTypedef	1469

Inherited Protected Data

None

IVGSequence

Class header file: iseq.h

IVGSequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequence
IVGSequence
IVSequence

IVGSequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSequence

Constructs an IVGSequence object.

Overload 1

```
public:IVGSequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSequenceTypedef

```
protected:IVGSequenceTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSequence < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2310	os_mark_schema_type	2310

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence	1475	operator =	1476

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence ImplOf	1475 1476	IGSequenceTypedef	1476

Inherited Protected Data

None

IVGSequenceAsDilTable

Class header file: iseqdil.h

IVGSequenceAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsDilTable
IVGSequenceAsDilTable
IVSequenceAsDilTable

IVGSequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSequenceAsDilTable

Constructs an IVGSequenceAsDilTable object.

Overload 1

```
public:IVGSequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Defintions

Protected type definitions associated with this class.

IVGSequenceAsDilTableTypedef

```
protected:IVGSequenceAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSequenceAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2314	os_mark_schema_type	2314

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable	1479	operator =	1480

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable ImplOf	1479 1481	IGSequenceAsDilTableTypedef	1480

Inherited Protected Data

None

IVGSequenceAsList

Class header file: iseqlst.h

IVGSequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsList
IVGSequenceAsList
IVSequenceAsList

IVGSequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSequenceAsList

Constructs an IVGSequenceAsList object.

Overload 1

```
public:IVGSequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSequenceAsListTypedef

```
protected:IVGSequenceAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSequenceAsList < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2318	os_mark_schema_type	2318

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList	1485	operator =	1486

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList ImplOf	1485 1487	IGSequenceAsListTypedef	1486

Inherited Protected Data

None

IVGSequenceAsTable

Class header file: iseqtab.h

IVGSequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsTable
IVGSequenceAsTable
IVSequenceAsTable

IVGSequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSequenceAsTable

Constructs an IVGSequenceAsTable object.

Overload 1

```
public:IVGSequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSequenceAsTableTypedef

```
protected:IVGSequenceAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSequenceAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2322	os_mark_schema_type	2322

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable	1491	operator =	1492

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable ImplOf	1491 1493	IGSequenceAsTableTypedef	1492

Inherited Protected Data

None

IVGSet

Class header file: iset.h

IVGSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSet
IVGSet
IVSet

IVGSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSet

Constructs an IVGSet object.

Overload 1

```
public:IVGSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetTypedef

```
protected:IVGSetTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSet < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2326	os_mark_schema_type	2326

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet	1499	operator =	1500

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet	1499	IGSetTypedef	1500
ImplOf	1500		

Inherited Protected Data

None

IVGSetAsAvlTree

Class header file: isetavl.h

IVGSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsAvlTree
IVGSetAsAvlTree
IVSetAsAvlTree

IVGSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsAvlTree

Constructs an IVGSetAsAvlTree object.

Overload 1

```
public:IVGSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsAvlTreeTypedef

```
protected:IVGSetAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsAvlTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2330	os_mark_schema_type	2330

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	operator =	1504

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	IGSetAsAvlTreeTypedef	1504
ImplOf	1504		

Inherited Protected Data

None

IVGSetAsBstTree

Class header file: isetbst.h

IVGSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsBstTree
IVGSetAsBstTree
IVSetAsBstTree

IVGSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsBstTree

Constructs an IVGSetAsBstTree object.

Overload 1

```
public:IVGSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsBstTreeTypedef

```
protected:IVGSetAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsBstTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2333	os_mark_schema_type	2333

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree	1509	operator =	1509

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree	1509	IGSetAsBstTreeTypedef	1509
ImplOf	1510		

Inherited Protected Data

None

IVGSetAsDilTable

Class header file: isetdil.h

IVGSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsDilTable
IVGSetAsDilTable
IVSetAsDilTable

IVGSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsDilTable

Constructs an IVGSetAsDilTable object.

Overload 1

```
public:IVGSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsDilTableTypedef

```
protected:IVGSetAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2337	os_mark_schema_type	2337

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	operator =	1514

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	IGSetAsDilTableTypedef	1515
ImplOf	1515		

Inherited Protected Data

None

IVGSetAsHshTable

Class header file: isethsh.h

IVGSetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsHshTable
IVGSetAsHshTable
IVSetAsHshTable

IVGSetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsHshTable

Constructs an IVGSetAsHshTable object.

Overload 1

```
public:IVGSetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsHshTableTypedef

```
protected:IVGSetAsHshTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsHshTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2340	os_mark_schema_type	2340

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable	1519	operator =	1520

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet numberOfOccurrences	1003 1009	locateNext removeAllOccurrences	1008 1009

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable ImplOf	1519 1521	IGSetAsHshTableTypedef	1520

Inherited Protected Data

None

IVGSetAsList

Class header file: isetlst.h

IVGSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsList
IVGSetAsList
IVSetAsList

IVGSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsList

Constructs an IVGSetAsList object.

Overload 1

```
public:IVGSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsListTypedef

```
protected:IVGSetAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsList < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2344	os_mark_schema_type	2344

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	operator =	1525

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	IGSetAsListTypedef	1525
ImplOf	1526		

Inherited Protected Data

None

IVGSetAsTable

Class header file: isettab.h

IVGSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsTable
IVGSetAsTable
IVSetAsTable

IVGSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSetAsTable

Constructs an IVGSetAsTable object.

Overload 1

```
public:IVGSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSetAsTableTypedef

```
protected:IVGSetAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSetImpl < IVSetImpl < PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSetAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2347	os_mark_schema_type	2347

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable	1530	operator =	1531

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet numberOfOccurrences	1003 1009	locateNext removeAllOccurrences	1008 1009

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable ImplOf	1530 1531	IGSetAsTableTypedef	1531

Inherited Protected Data

None

IVGSortedBag

Class header file: isb.h

IVGSortedBag - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBag
IVGSortedBag
IVSortedBag

IVGSortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBag

Constructs an IVGSortedBag object.

Overload 1

```
public:IVGSortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagTypedef

```
protected:IVGSortedBagTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBag < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2351	os_mark_schema_type	2351

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag	1537	operator =	1537

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag ImplOf	1537 1538	IGSortedBagTypedef	1538

Inherited Protected Data

None

IVGSortedBagAsAvlTree

Class header file: isbavl.h

IVGSortedBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsAvlTree
IVGSortedBagAsAvlTree
IVSortedBagAsAvlTree

IVGSortedBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBagAsAvlTree

Constructs an IVGSortedBagAsAvlTree object.

Overload 1

```
public:IVGSortedBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagAsAvlTreeTypedef

```
protected:IVGSortedBagAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBagAsAvlTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2355	os_mark_schema_type	2355

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree	1541	operator =	1542

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree ImplOf	1541 1543	IGSortedBagAsAvlTreeTypedef	1542

Inherited Protected Data

None

IVGSortedBagAsBstTree

Class header file: isbbst.h

IVGSortedBagAsBstTree - Hierarchy List

IMetaCollection

IACollectionBase

IACollection

IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsBstTree
IVGSortedBagAsBstTree
IVSortedBagAsBstTree

IVGSortedBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBagAsBstTree

Constructs an IVGSortedBagAsBstTree object.

Overload 1

```
public:IVGSortedBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagAsBstTreeTypedef

```
protected:IVGSortedBagAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBagAsBstTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2359	os_mark_schema_type	2359

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree	1547	operator =	1548

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree ImplOf	1547 1549	IGSortedBagAsBstTreeTypedef	1548

Inherited Protected Data

None

IVGSortedBagAsDilTable

Class header file: isbdil.h

IVGSortedBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsDilTable
IVGSortedBagAsDilTable
IVSortedBagAsDilTable

IVGSortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBagAsDilTable

Constructs an IVGSortedBagAsDilTable object.

Overload 1

```
public:IVGSortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagAsDilTableTypedef

```
protected:IVGSortedBagAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBagAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2362	os_mark_schema_type	2363

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable	1554	operator =	1554

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable ImplOf	1554 1555	IGSortedBagAsDilTableTypedef	1554

Inherited Protected Data

None

IVGSortedBagAsList

Class header file: isblst.h

IVGSortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsList
IVGSortedBagAsList
IVSortedBagAsList

IVGSortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBagAsList

Constructs an IVGSortedBagAsList object.

Overload 1

```
public:IVGSortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagAsListTypedef

```
protected:IVGSortedBagAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBagAsList < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2366	os_mark_schema_type	2366

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList	1560	operator =	1560

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList ImplOf	1560 1561	IGSortedBagAsListTypedef	1560

Inherited Protected Data

None

IVGSortedBagAsTable

Class header file: isbtab.h

IVGSortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsTable
IVGSortedBagAsTable
IVSortedBagAsTable

IVGSortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedBagAsTable

Constructs an IVGSortedBagAsTable object.

Overload 1

```
public:IVGSortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedBagAsTableTypedef

```
protected:IVGSortedBagAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedBagImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedBagImpl < IVSortedBagImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedBagAsTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2370	os_mark_schema_type	2370

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable	1566	operator =	1566

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable ImplOf	1566 1567	IGSortedBagAsTableTypedef	1567

Inherited Protected Data

None

IVGSortedMap

Variant of ISortedMap enabled for notifications.

Class header file: ism.h

IVGSortedMap - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMap
IVGSortedMap
IVSortedMap

IVGSortedMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMap

Constructs an IVGSortedMap object.

Overload 1

```
public:IVGSortedMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMap < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2374	os_mark_schema_type	2374

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	operator =	1574

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	IGSortedMapTypedef	1574
ImplOf	1575		

IVGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapTypedef	2373		

Inherited Protected Data

None

IVGSortedMapAsAvlTree

Class header file: ismavl.h

IVGSortedMapAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsAvlTree
IVGSortedMapAsAvlTree
IVSortedMapAsAvlTree

IVGSortedMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMapAsAvlTree

Constructs an IVGSortedMapAsAvlTree object.

Overload 1

```
public:IVGSortedMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedMapAsAvlTreeTypedef

```
protected:IVGSortedMapAsAvlTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMapAsAvlTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2378	os_mark_schema_type	2378

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	operator =	1579

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	IGSortedMapAsAvlTreeTypedef	1579
ImplOf	1580		

Inherited Protected Data

None

IVGSortedMapAsBstTree

Class header file: ismbst.h

IVGSortedMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsBstTree
IVGSortedMapAsBstTree
IVSortedMapAsBstTree

IVGSortedMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMapAsBstTree

Constructs an IVGSortedMapAsBstTree object.

Overload 1

```
public:IVGSortedMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedMapAsBstTreeTypedef

```
protected:IVGSortedMapAsBstTreeTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMapAsBstTree < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2382	os_mark_schema_type	2382

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	operator =	1585

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	IGSortedMapAsBstTreeTypedef	1585
ImplOf	1586		

Inherited Protected Data

None

IVGSortedMapAsDilTable

Class header file: ismdil.h

IVGSortedMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap

IGSortedMapAsDilTable
IVGSortedMapAsDilTable
IVSortedMapAsDilTable

IVGSortedMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMapAsDilTable

Constructs an IVGSortedMapAsDilTable object.

Overload 1

```
public:IVGSortedMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedMapAsDilTableTypedef

```
protected:IVGSortedMapAsDilTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMapAsDilTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2386	os_mark_schema_type	2386

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	operator =	1591

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	IGSortedMapAsDilTableTypedef	1591
ImplOf	1592		

Inherited Protected Data

None

IVGSortedMapAsList

Class header file: ismlst.h

IVGSortedMapAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsList
IVGSortedMapAsList
IVSortedMapAsList

IVGSortedMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMapAsList

Constructs an IVGSortedMapAsList object.

Overload 1

```
public:IVGSortedMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedMapAsListTypedef

```
protected:IVGSortedMapAsListTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMapAsList < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2389	os_mark_schema_type	2389

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	operator =	1597

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	IGSortedMapAsListTypedef	1597
ImplOf	1598		

Inherited Protected Data

None

IVGSortedMapAsTable

Class header file: ismtab.h

IVGSortedMapAsTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - IGSortedMapAsTable
            - **IVGSortedMapAsTable**
              - IVSortedMapAsTable

IVGSortedMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedMapAsTable

Constructs an IVGSortedMapAsTable object.

Overload 1

```
public:IVGSortedMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedMapAsTableTypedef

```
protected:IVGSortedMapAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedMapImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedMapImpl < IVSortedMapImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedMapAsTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

IVGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2393	os_mark_schema_type	2393

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsTable	1603	operator =	1603

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsTable	1603	IGSortedMapAsTableTypedef	1604
ImplOf	1604		

Inherited Protected Data

None

IVGSortedRelation

Class header file: isr.h

IVGSortedRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelation
IVGSortedRelation
IVSortedRelation

IVGSortedRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedRelation

Constructs an IVGSortedRelation object.

Overload 1

```
public:IVGSortedRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is

unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedRelationTypedef

```
protected:IVGSortedRelationTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedRelationImpl < IVSortedRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedRelation < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2397	os_mark_schema_type	2397

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation	1611	operator =	1611

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation ImplOf	1611 1612	IGSortedRelationTypedef	1612

Inherited Protected Data

None

IVGSortedRelationAsDilTable

Class header file: isrdil.h

IVGSortedRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsDilTable
IVGSortedRelationAsDilTable
IVSortedRelationAsDilTable

IVGSortedRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IVGSortedRelationAsDilTable

Constructs an IVGSortedRelationAsDilTable object.

Overload 1

```
public:IVGSortedRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.
IVGSortedRelationAsDilTableTypedef

```
protected:IVGSortedRelationAsDilTableTypedef( Element,
                                              Key, ElementOps )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedRelationImpl < IVSortedRelationImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedRelationAsDilTable < Element , Key ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2401	os_mark_schema_type	2401

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable	1615	operator =	1616

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable ImplOf	1615 1617	IGSortedRelationAsDilTableTypedef	1616

Inherited Protected Data

None

IVGSortedRelationAsTable

Class header file: isrtab.h

IVGSortedRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsTable
IVGSortedRelationAsTable
IVSortedRelationAsTable

IVGSortedRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedRelationAsTable

Constructs an IVGSortedRelationAsTable object.

Overload 1

```
public:IVGSortedRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedRelationAsTableTypedef

```
protected:IVGSortedRelationAsTableTypedef(Element, Key, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedRelationImpl < IVSortedRelationImpl <
        PureImplementation > > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedRelationImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedRelationAsTable < Element , Key ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2405	os_mark_schema_type	2405

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable	1621	operator =	1622

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable ImplOf	1621 1623	IGSortedRelationAsTableTypedef	1622

Inherited Protected Data

None

IVGSortedSet

The default implementation, notification-enabled variant of ISortedSet.

Class header file: iss.h

IVGSortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSet
IVGSortedSet
IVSortedSet

IVGSortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSet

Constructs an IVGSortedSet object.

Overload 1

```
public:IVGSortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetTypedef

```
protected:IVGSortedSetTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct an IVGSortedSet object.

IVGSortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Implementation

This member is not accessible.

Variation 1

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSet < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2409	os_mark_schema_type	2409

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet	1629	operator =	1630

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet ImplOf	1629 1630	IGSortedSetTypedef	1629

Inherited Protected Data

None

IVGSortedSetAsAvlTree

Class header file: issavl.h

IVGSortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsAvlTree
IVGSortedSetAsAvlTree

IVGSortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSetAsAvlTree

Constructs an IVGSortedSetAsAvlTree object.

Overload 1

```
public:IVGSortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedSetAsAvlTreeTypedef

```
protected:IVGSortedSetAsAvlTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSetAsAvlTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2413	os_mark_schema_type	2413

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree	1633	operator =	1634

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree ImplOf	1633 1635	IGSortedSetAsAvlTreeTypedef	1634

Inherited Protected Data

None

IVGSortedSetAsBstTree

Class header file: issbst.h

IVGSortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsBstTree
IVGSortedSetAsBstTree
IVSortedSetAsBstTree

IVGSortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSetAsBstTree

Constructs an IVGSortedSetAsBstTree object.

Overload 1

```
public:IVGSortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedSetAsBstTreeTypedef

```
protected:IVGSortedSetAsBstTreeTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSetAsBstTree < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2416	os_mark_schema_type	2417

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree	1639	operator =	1640

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree ImplOf	1639 1641	IGSortedSetAsBstTreeTypedef	1640

Inherited Protected Data

None

IVGSortedSetAsDilTable

Class header file: issdil.h

IVGSortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsDilTable
IVGSortedSetAsDilTable
IVSortedSetAsDilTable

IVGSortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSetAsDilTable

Constructs an IVGSortedSetAsDilTable object.

Overload 1

```
public:IVGSortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedSetAsDilTableTypedef

```
protected:IVGSortedSetAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSetAsDilTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2420	os_mark_schema_type	2420

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable	1645	operator =	1646

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable ImplOf	1645 1647	IGSortedSetAsDilTableTypedef	1646

Inherited Protected Data

None

IVGSortedSetAsList

Class header file: isslst.h

IVGSortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsList
IVGSortedSetAsList
IVSortedSetAsList

IVGSortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSetAsList

Constructs an IVGSortedSetAsList object.

Overload 1

```
public:IVGSortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedSetAsListTypedef

```
protected:IVGSortedSetAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSetAsList < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2424	os_mark_schema_type	2424

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList	1652	operator =	1652

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList ImplOf	1652 1653	IGSortedSetAsListTypedef	1652

Inherited Protected Data

None

IVGSortedSetAsTable

Class header file: isstab.h

IVGSortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsTable
IVGSortedSetAsTable
IVSortedSetAsTable

IVGSortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGSortedSetAsTable

Constructs an IvGSortedSetAsTable object.

Overload 1

```
public:IVGSortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGSortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGSortedSetAsTableTypedef

```
protected:IVGSortedSetAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef IVSortedSetImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef ICSortedSetImpl < IVSortedSetImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGSortedSetAsTable < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGSortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2428	os_mark_schema_type	2428

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable	1658	operator =	1658

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet	1041	numberOfOccurrences	1046
removeAllOccurrences	1046		

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable	1658	IGSortedSetAsTableTypedef	1658
ImplOf	1659		

Inherited Protected Data

None

IVGStack

Class header file: istk.h

IVGStack - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStack
IVGStack
IVStack

IVGStack - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGStack

Constructs an IVGStack object.

Overload 1

```
public:IVGStack(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGStack(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGStackTypedef

```
protected:IVGStackTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStack - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGStack < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStack - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVGStack			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2432	os_mark_schema_type	2432

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IGStack	1665	operator =	1666

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IGStack ImplOf	1665 1667	IGStackTypedef	1666

Inherited Protected Data

None

IVGStackAsDilTable

Class header file: istkdil.h

IVGStackAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsDilTable
IVGStackAsDilTable
IVStackAsDilTable

IVGStackAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGStackAsDilTable

Constructs an IVGStackAsDilTable object.

Overload 1

```
public:IVGStackAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGStackAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGStackAsDilTableTypedef

```
protected:IVGStackAsDilTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGStackAsDilTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2435	os_mark_schema_type	2435

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable	1669	operator =	1669

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable	1669	IGStackAsDilTableTypedef	1670
ImplOf	1670		

Inherited Protected Data

None

IVGStackAsList

Class header file: istklst.h

IVGStackAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsList
IVGStackAsList
IVStackAsList

IVGStackAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IVGStackAsList

Constructs an IVGStackAsList object.

Overload 1

```
public:IVGStackAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGStackAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGStackAsListTypedef

```
protected:IVGStackAsListTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGStackAsList < Element ,
        ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsList - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2438	os_mark_schema_type	2438

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	operator =	1675

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	IGStackAsListTypedef	1675
ImplOf	1676		

Inherited Protected Data

None

IVGStackAsTable

Class header file: istktab.h

IVGStackAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAStack
        IGStackAsTable

IVGStackAsTable

IVStackAsTable

IVGStackAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVGStackAsTable

Constructs an IVGStackAsTable object.

Overload 1

```
public:IVGStackAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVGStackAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected Type Definitions

Protected type definitions associated with this class.

IVGStackAsTableTypedef

```
protected:IVGStackAsTableTypedef(Element, ElementOps)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implementation

Variation 1

```
typedef ICSequenceImpl < IVSequenceImpl < PureImplementation >
        > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
typedef IVSequenceImpl < PureImplementation > Implementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME IGStackAsTable < Element ,
ElementOps >::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVGStackAsTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2442	os_mark_schema_type	2442

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable	1680	operator =	1680

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable	1680	IGStackAsTableTypedef	1680
ImplOf	1681		

Inherited Protected Data

None

IVHeap

Variant of IHeap enabled for notifications.

Class header file: ihp.h

IVHeap - Hierarchy List

IGHeap
  IVGHeap
    IVHeap

IVHeap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVHeap

Constructs an IVHeap object.

Overload 1

```
public:IVHeap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVHeap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVHeap < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeap - Inherited Member Functions and Data

Inherited Public Functions

IVHeap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2444	os_mark_schema_type	2444

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap	1216	operator =	1217

IVGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeap	2147		

Inherited Public Data

None

Inherited Protected Functions

IGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeap ImplOf	1216 1218	IGHeapTypedef	1217

IVGHeap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeap	2147	IVGHeapTypedef	2147

Inherited Protected Data

None

IVHeapAsDilTable

Variant of IHeapAsDilTable enabled for notifications.

Class header file: ihpdil.h

IVHeapAsDilTable - Hierarchy List

IGHeapAsDilTable
IVGHeapAsDilTable
IVHeapAsDilTable

IVHeapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVHeapAsDilTable

Constructs an IVHeapAsDilTable object.

Overload 1

```
public:IVHeapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVHeapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVHeapAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IVHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2446	os_mark_schema_type	2446

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable	1219	operator =	1219

IVGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsDilTable	2149		

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsDilTable ImplOf	1219 1220	IGHeapAsDilTableTypedef	1219

IVGHeapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsDilTable	2149	IVGHeapAsDilTableTypedef	2149

Inherited Protected Data

None

IVHeapAsList

Variant of IHeapAsList enabled for notifications.

Class header file: ihplst.h

IVHeapAsList - Hierarchy List

IGHeapAsList
  IVGHeapAsList

IVHeapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVHeapAsList

Constructs an IVHeapAsList object.

Overload 1

```
public:IVHeapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVHeapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVHeapAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsList - Inherited Member Functions and Data

Inherited Public Functions

IVHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2448	os_mark_schema_type	2448

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList	1222	operator =	1223

IVGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsList	2150		

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsList ImplOf	1222 1224	IGHeapAsListTypedef	1223

IVGHeapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsList	2150	IVGHeapAsListTypedef	2151

Inherited Protected Data

None

IVHeapAsTable

Variant of IHeapAsTable enabled for notifications.

Class header file: ihptab.h

IVHeapAsTable - Hierarchy List

IGHeapAsTable
  IVGHeapAsTable

IVHeapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVHeapAsTable

Constructs an IVHeapAsTable object.

Overload 1

```
public:IVHeapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVHeapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVHeapAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVHeapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IVHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2450	os_mark_schema_type	2450

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsTable	1226	operator =	1227

IVGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsTable	2152		

Inherited Public Data

None

Inherited Protected Functions

IGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGHeapAsTable ImplOf	1226 1227	IGHeapAsTableTypedef	1227

IVGHeapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGHeapAsTable	2152	IVGHeapAsTableTypedef	2153

Inherited Protected Data

None

IVKeyBag

Variant of IKeyBag enabled for notifications.

Class header file: ikb.h

IVKeyBag - Hierarchy List

IMetaCollection

IACollectionBase

IACollection
IAKeyCollection
IAKeyBag
IGKeyBag
IVGKeyBag
IVKeyBag

IVKeyBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeyBag

Constructs an IVKeyBag object.

Overload 1

```
public:IVKeyBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeyBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeyBag < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2452	os_mark_schema_type	2452

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBag	1231	operator =	1232

IVGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBag	2154		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBag ImplOf	1231 1233	IGKeyBagTypedef	1232

IVGKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBag	2154	IVGKeyBagTypedef	2155

Inherited Protected Data

None

IVKeyBagAsDilTable

Variant of IKeyBagAsDilTable enabled for notifications.

Class header file: ikbdil.h

IVKeyBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsDilTable
IVGKeyBagAsDilTable
IVKeyBagAsDilTable

IVKeyBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeyBagAsDilTable

Constructs an IVKeyBagAsDilTable object.

Overload 1

```
public:IVKeyBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeyBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeyBagAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2456	os_mark_schema_type	2456

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable	1235	operator =	1236

IVGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsDilTable	2158		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsDilTable ImplOf	1235 1236	IGKeyBagAsDilTableTypedef	1236

IVGKeyBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsDilTable	2158	IVGKeyBagAsDilTableTypedef	2158

Inherited Protected Data

None

IVKeyBagAsHshTable

Variant of IKeyBagAsHshTable enabled for notifications.

Class header file: ikbhsh.h

IVKeyBagAsHshTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeyBag
        - IGKeyBagAsHshTable
          - IVGKeyBagAsHshTable

IVKeyBagAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeyBagAsHshTable

Constructs an IVKeyBagAsHshTable object.

Overload 1

```
public:IVKeyBagAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeyBagAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeyBagAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2459	os_mark_schema_type	2459

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable	1240	operator =	1241

IVGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsHshTable	2161		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsHshTable ImplOf	1240 1242	IGKeyBagAsHshTableTypedef	1241

IVGKeyBagAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsHshTable	2161	IVGKeyBagAsHshTableTypedef	2161

Inherited Protected Data

None

IVKeyBagAsList

Variant of IKeyBagAsList enabled for notifications.

Class header file: ikblst.h

IVKeyBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeyBag
IGKeyBagAsList
IVGKeyBagAsList
IVKeyBagAsList

IVKeyBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeyBagAsList

Constructs an IVKeyBagAsList object.

Overload 1

```
public:IVKeyBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeyBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeyBagAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2463	os_mark_schema_type	2463

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList	1246	operator =	1246

IVGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsList	2164		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsList ImplOf	1246 1247	IGKeyBagAsListTypedef	1246

IVGKeyBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsList	2164	IVGKeyBagAsListTypedef	2165

Inherited Protected Data

None

IVKeyBagAsTable

Variant of IKeyBagAsList enabled for notifications.

Class header file: ikbtab.h

IVKeyBagAsTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeyBag
        - IGKeyBagAsTable
          - IVGKeyBagAsTable

IVKeyBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeyBagAsTable

Constructs an IVKeyBagAsTable object.

Overload 1

```
public:IVKeyBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeyBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeyBagAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeyBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyBag	906	IAKeyBag	906

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IVKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2466	os_mark_schema_type	2466

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable	1251	operator =	1252

IVGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsTable	2168		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyBag	906		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeyBagAsTable ImplOf	1251 1253	IGKeyBagAsTableTypedef	1252

IVGKeyBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeyBagAsTable	2168	IVGKeyBagAsTableTypedef	2168

Inherited Protected Data

None

IVKeySet

Variant of IKeySet enabled for notifications.

Class header file: iks.h

IVKeySet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySet
IVGKeySet
IVKeySet

IVKeySet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySet

Constructs an IVKeySet object.

Overload 1

```
public:IVKeySet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySet < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IVKeySet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2470	os_mark_schema_type	2470

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	operator =	1259

IVGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySet	2171		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySet	1258	IGKeySetTypedef	1258
ImplOf	1260		

IVGKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySet	2171	IVGKeySetTypedef	2171

Inherited Protected Data

None

IVKeySetAsAvlTree

Variant of IKeySetAsAvlTree enabled for notifications.

Class header file: iksavl.h

IVKeySetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsAvlTree
IVGKeySetAsAvlTree
IVKeySetAsAvlTree

IVKeySetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsAvlTree

Constructs an IVKeySetAsAvlTree object.

Overload 1

```
public:IVKeySetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2473	os_mark_schema_type	2473

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree	1262	operator =	1263

IVGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsAvlTree	2175		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsAvlTree ImplOf	1262 1264	IGKeySetAsAvlTreeTypedef	1263

IVGKeySetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsAvlTree	2175	IVGKeySetAsAvlTreeTypedef	2175

Inherited Protected Data

None

IVKeySetAsBstTree

Class header file: iksbst.h

IVKeySetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsBstTree
IVGKeySetAsBstTree
IVKeySetAsBstTree

IVKeySetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsBstTree

Constructs an IVKeySetAsBstTree object.

Overload 1

```
public:IVKeySetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2477	os_mark_schema_type	2477

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree	1268	operator =	1268

IVGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsBstTree	2178		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsBstTree	1268	IGKeySetAsBstTreeTypedef	1268
ImplOf	1269		

IVGKeySetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsBstTree	2178	IVGKeySetAsBstTreeTypedef	2179

Inherited Protected Data

None

IVKeySetAsDilTable

Variant of IKeySetAsDilTable enabled for notifications.

Class header file: iksdil.h

IVKeySetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsDilTable

IVKeySetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsDilTable

Constructs an IVKeySetAsDilTable object.

Overload 1

```
public:IVKeySetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2480	os_mark_schema_type	2480

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	operator =	1274

IVGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsDilTable	2182		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsDilTable	1273	IGKeySetAsDilTableTypedef	1274
ImplOf	1275		

IVGKeySetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsDilTable	2182	IVGKeySetAsDilTableTypedef	2182

Inherited Protected Data

None

IVKeySetAsHshTable

Variant of IKeySetAsHshTable enabled for notifications.

Class header file: ikshsh.h

IVKeySetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsHshTable
IVGKeySetAsHshTable
IVKeySetAsHshTable

IVKeySetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsHshTable

Constructs an IVKeySetAsHshTable object.

Overload 1

```
public:IVKeySetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2484	os_mark_schema_type	2484

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable	1279	operator =	1279

IVGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsHshTable	2185		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsHshTable	1279	IGKeySetAsHshTableTypedef	1279
ImplOf	1280		

IVGKeySetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsHshTable	2185	IVGKeySetAsHshTableTypedef	2186

Inherited Protected Data

None

IVKeySetAsList

Variant of IKeySetAsList enabled for notifications.

Class header file: ikslst.h

IVKeySetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsList

IVKeySetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsList

Constructs an IVKeySetAsList object.

Overload 1

```
public:IVKeySetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2487	os_mark_schema_type	2487

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	operator =	1285

IVGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsList	2189		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsList	1284	IGKeySetAsListTypedef	1285
ImplOf	1286		

IVGKeySetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsList	2189	IVGKeySetAsListTypedef	2189

Inherited Protected Data

None

IVKeySetAsTable

Variant of IKeySetAsTable enabled for notifications.

Class header file: ikstab.h

IVKeySetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySet
IGKeySetAsTable
IVGKeySetAsTable
IVKeySetAsTable

IVKeySetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySetAsTable

Constructs an IVKeySetAsTable object.

Overload 1

```
public:IVKeySetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySetAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateOrAddElementWithKey	912	removeAllElementsWithKey	913
removeElementWithKey	914	replaceElementWithKey	914

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySet	917	IAKeySet	917

IVKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2491	os_mark_schema_type	2491

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable	1290	operator =	1290

IVGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsTable	2192		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySet	917	locateNextElementWithKey	918
numberOfDifferentKeys	918	numberOfElementsWithKey	918
removeAllElementsWithKey	918	setToNextWithDifferentKey	919

IGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySetAsTable	1290	IGKeySetAsTableTypedef	1290
ImplOf	1291		

IVGKeySetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySetAsTable	2192	IVGKeySetAsTableTypedef	2193

Inherited Protected Data

None

IVKeySortedBag

Variant of IKeySortedBag enabled for notifications.

Class header file: iksb.h

IVKeySortedBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeySortedCollection
        - IAKeySortedBag

IGKeySortedBag
IVGKeySortedBag
IVKeySortedBag

IVKeySortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedBag

Constructs an IVKeySortedBag object.

Overload 1

```
public:IVKeySortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

Self

```
typedef IVKeySortedBag < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IVKeySortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2494	os_mark_schema_type	2494

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag	1297	operator =	1297

IVGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBag	2196		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBag ImplOf	1297 1298	IGKeySortedBagTypedef	1296

IVGKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBag	2196	IVGKeySortedBagTypedef	2196

Inherited Protected Data

None

IVKeySortedBagAsDilTable

Variant of IKeySortedBagAsDilTable enabled for notifications.

Class header file: iksbdil.h

IVKeySortedBagAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsDilTable
IVGKeySortedBagAsDilTable
IVKeySortedBagAsDilTable

IVKeySortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedBagAsDilTable

Constructs an IVKeySortedBagAsDilTable object.

Overload 1

```
public:IVKeySortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedBagAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2498	os_mark_schema_type	2498

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable	1301	operator =	1302

IVGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsDilTable	2200		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsDilTable ImplOf	1301 1303	IGKeySortedBagAsDilTableTypedef	1302

IVGKeySortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsDilTable	2200	IVGKeySortedBagAsDilTableTypedef	2200

Inherited Protected Data

None

IVKeySortedBagAsList

Variant of IKeySortedBagAsList enabled for notifications.

Class header file: iksblst.h

IVKeySortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedBag
IGKeySortedBagAsList
IVGKeySortedBagAsList
IVKeySortedBagAsList

IVKeySortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedBagAsList

Constructs an IVKeySortedBagAsList object.

Overload 1

```
public:IVKeySortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedBagAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2502	os_mark_schema_type	2502

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList	1307	operator =	1308

IVGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsList	2204		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsList ImplOf	1307 1309	IGKeySortedBagAsListTypedef	1308

IVGKeySortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsList	2204	IVGKeySortedBagAsListTypedef	2204

Inherited Protected Data

None

IVKeySortedBagAsTable

Variant of IKeySortedBagAsTable enabled for notifications.

Class header file: iksbtab.h

IVKeySortedBagAsTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeySortedCollection
        - IAKeySortedBag
          - IGKeySortedBagAsTable
            - IVGKeySortedBagAsTable
              - **IVKeySortedBagAsTable**

IVKeySortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedBagAsTable

Constructs an IVKeySortedBagAsTable object.

Overload 1

```
public:IVKeySortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedBagAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
numberOfDifferentKeys	913	numberOfElementsWithKey	913
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914	setToNextWithDifferentKey	914

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedBag	921	compare	923
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IVKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2506	os_mark_schema_type	2506

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable	1313	operator =	1314

IVGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsTable	2207		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedBag	922		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedBagAsTable ImplOf	1313 1315	IGKeySortedBagAsTableTypedef	1314

IVGKeySortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedBagAsTable	2207	IVGKeySortedBagAsTableTypedef	2208

Inherited Protected Data

None

IVKeySortedSet

Variant of IKeySortedSet enabled for notifications.

Class header file: ikss.h

IVKeySortedSet - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAKeyCollection
      - IAKeySortedCollection
        - IAKeySortedSet
          - IGKeySortedSet
            - IVGKeySortedSet
              - **IVKeySortedSet**

IVKeySortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSet

Constructs an IVKeySortedSet object.

Overload 1

```
public:IVKeySortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSet < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2510	os_mark_schema_type	2510

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet	1321	operator =	1322

IVGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSet	2211		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSet ImplOf	1321 1323	IGKeySortedSetTypedef	1322

IVGKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSet	2211	IVGKeySortedSetTypedef	2212

Inherited Protected Data

None

IVKeySortedSetAsAvlTree

Variant of IKeySortedSetAsAvlTree enabled for notifications.

Class header file: ikssavl.h

IVKeySortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsAvlTree
IVGKeySortedSetAsAvlTree
IVKeySortedSetAsAvlTree

IVKeySortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSetAsAvlTree

Constructs an IVKeySortedSetAsAvlTree object.

Overload 1

```
public:IVKeySortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSetAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2514	os_mark_schema_type	2514

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree	1326	operator =	1326

IVGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsAvlTree	2215		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsAvlTree ImplOf	1326 1327	IGKeySortedSetAsAvlTreeTypedef	1326

IVGKeySortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsAvlTree	2215	IVGKeySortedSetAsAvlTreeTypedef	2215

Inherited Protected Data

None

IVKeySortedSetAsBstTree

Variant of IKeySortedSetAsBstTree enabled for notifications.

Class header file: ikssbst.h

IVKeySortedSetAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsBstTree
IVGKeySortedSetAsBstTree
IVKeySortedSetAsBstTree

IVKeySortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSetAsBstTree

Constructs an IVKeySortedSetAsBstTree object.

Overload 1

```
public:IVKeySortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSetAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2518	os_mark_schema_type	2518

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree	1332	operator =	1332

IVGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsBstTree	2219		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsBstTree	1332	IGKeySortedSetAsBstTreeTypedef	1332
ImplOf	1333		

IVGKeySortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGKeySortedSetAsBstTree`	2219	`IVGKeySortedSetAsBstTreeTypedef`	2219

Inherited Protected Data

None

IVKeySortedSetAsDilTable

Variant of IKeySortedSetAsDilTable enabled for notifications.

Class header file: ikssdil.h

IVKeySortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsDilTable
IVGKeySortedSetAsDilTable
IVKeySortedSetAsDilTable

IVKeySortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSetAsDilTable

Constructs an IVKeySortedSetAsDilTable object.

Overload 1

```
public:IVKeySortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSetAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet	929	compare	930
IAKeySortedSet	929		

IVKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2522	os_mark_schema_type	2522

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable	1338	operator =	1338

IVGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsDilTable	2223		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet	929	numberOfDifferentKeys	931
numberOfElementsWithKey	931	removeAllElementsWithKey	931
setToNextWithDifferentKey	931		

IGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsDilTable	1338	IGKeySortedSetAsDilTableTypedef	1338
ImplOf	1339		

IVGKeySortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsDilTable	2223	IVGKeySortedSetAsDilTableTypedef	2223

Inherited Protected Data

None

IVKeySortedSetAsList

Variant of IKeySortedSetAsList enabled for notifications.

Class header file: iksslst.h

IVKeySortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection
IAKeySortedSet
IGKeySortedSetAsList
IVGKeySortedSetAsList
IVKeySortedSetAsList

IVKeySortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSetAsList

Constructs an IVKeySortedSetAsList object.

Overload 1

```
public:IVKeySortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSetAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IVKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2526	os_mark_schema_type	2526

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList	1344	operator =	1344

IVGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsList	2226		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsList ImplOf	1344 1345	IGKeySortedSetAsListTypedef	1344

IVGKeySortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsList	2226	IVGKeySortedSetAsListTypedef	2227

Inherited Protected Data

None

IVKeySortedSetAsTable

Variant of IKeySortedSetAsTable enabled for notifications.

Class header file: iksstab.h

IVKeySortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAKeyCollection
IAKeySortedCollection

IAKeySortedSet
IGKeySortedSetAsTable
IVGKeySortedSetAsTable
IVKeySortedSetAsTable

IVKeySortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVKeySortedSetAsTable

Constructs an IVKeySortedSetAsTable object.

Overload 1

```
public:IVKeySortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVKeySortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVKeySortedSetAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVKeySortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeyCollection	909	addOrReplaceElementWithKey	910
containsAllKeysFrom	911	containsElementWithKey	911
elementWithKey	911	IAKeyCollection	909
key	911	locateElementWithKey	912
locateNextElementWithKey	912	locateOrAddElementWithKey	912
removeAllElementsWithKey	913	removeElementWithKey	914
replaceElementWithKey	914		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedCollection	925	IAKeySortedCollection	926

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IAKeySortedSet IAKeySortedSet	929 929	compare	930

IVKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2529	os_mark_schema_type	2529

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable	1350	operator =	1350

IVGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsTable	2230		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeyCollection	909		

IAKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedCollection	926		

IAKeySortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IAKeySortedSet numberOfElementsWithKey setToNextWithDifferentKey	929 931 931	numberOfDifferentKeys removeAllElementsWithKey	931 931

IGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGKeySortedSetAsTable ImplOf	1350 1351	IGKeySortedSetAsTableTypedef	1350

IVGKeySortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGKeySortedSetAsTable	2230	IVGKeySortedSetAsTableTypedef	2231

Inherited Protected Data

None

IVMap

Variant of IMap enabled for notifications.

Class header file: imap.h

IVMap - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IAMap
          - IGMap
            - IVGMap
              - **IVMap**

IVMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMap

Constructs an IVMap object.

Overload 1

```
public:IVMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMap < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2533	os_mark_schema_type	2533

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGMap	1357	operator =	1358

IVGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMap	2234		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGMap	1357	IGMapTypedef	1358
ImplOf	1359		

IVGMap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMap	2234	IVGMapTypedef	2235

Inherited Protected Data

None

IVMapAsAvlTree

Variant of IMapAsAvlTree enabled for notifications.

Class header file: imapavl.h

IVMapAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IAMap
          - IGMapAsAvlTree
            - IVGMapAsAvlTree
              - **IVMapAsAvlTree**

IVMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsAvlTree

Constructs an IVMapAsAvlTree object.

Overload 1

```
public:IVMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2537	os_mark_schema_type	2537

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree	1362	operator =	1362

IVGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsAvlTree	2238		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	941		

IGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsAvlTree ImplOf	1362 1363	IGMapAsAvlTreeTypedef	1362

IVGMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsAvlTree	2238	IVGMapAsAvlTreeTypedef	2238

Inherited Protected Data

None

IVMapAsBstTree

Variant of IMapAsBstTree enabled for notifications.

Class header file: imapbst.h

IVMapAsBstTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeyCollection
        - IAMap
          - IGMapAsBstTree
            - IVGMapAsBstTree
              - **IVMapAsBstTree**

IVMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsBstTree

Constructs an IVMapAsBstTree object.

Overload 1

```
public:IVMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2541	os_mark_schema_type	2541

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsBstTree	1367	operator =	1368

IVGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsBstTree	2242		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap locateNextElementWithKey numberOfElementsWithKey removeAllElementsWithKey setToNextWithDifferentKey	934 939 940 940 941	locateNext numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	939 940 940 941

IGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGMapAsBstTree` `ImplOf`	1367 1369	`IGMapAsBstTreeTypedef`	1368

IVGMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGMapAsBstTree`	2242	`IVGMapAsBstTreeTypedef`	2242

Inherited Protected Data

None

IVMapAsDilTable

Variant of IMapAsDilTable enabled for notifications.

Class header file: imapdil.h

IVMapAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsDilTable
IVGMapAsDilTable
IVMapAsDilTable

IVMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsDilTable

Constructs an IVMapAsDilTable object.

Overload 1

```
public:IVMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2545	os_mark_schema_type	2545

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsDilTable	1373	operator =	1374

IVGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsDilTable	2246		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsDilTable	1373	IGMapAsDilTableTypedef	1374
ImplOf	1374		

IVGMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsDilTable	2246	IVGMapAsDilTableTypedef	2246

Inherited Protected Data

None

IVMapAsHshTable

Variant of IMapAsHshTable enabled for notifications.

Class header file: imaphsh.h

IVMapAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsHshTable
IVGMapAsHshTable
IVMapAsHshTable

IVMapAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsHshTable

Constructs an IVMapAsHshTable object.

Overload 1

```
public:IVMapAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2549	os_mark_schema_type	2549

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable	1379	operator =	1379

IVGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsHshTable	2249		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsHshTable	1379	IGMapAsHshTableTypedef	1379
ImplOf	1380		

IVGMapAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGMapAsHshTable`	2249	`IVGMapAsHshTableTypedef`	2250

Inherited Protected Data

None

IVMapAsList

Variant of IMapAsList enabled for notifications.

Class header file: imaplst.h

IVMapAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsList
IVGMapAsList
IVMapAsList

IVMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsList

Constructs an IVMapAsList object.

Overload 1

```
public:IVMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2553	os_mark_schema_type	2553

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList	1385	operator =	1385

IVGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsList	2253		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsList	1385	IGMapAsListTypedef	1385
ImplOf	1386		

IVGMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsList	2253	IVGMapAsListTypedef	2254

Inherited Protected Data

None

IVMapAsTable

Variant of IMapAsTable enabled for notifications.

Class header file: imaptab.h

IVMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IAMap
IGMapAsTable
IVGMapAsTable
IVMapAsTable

IVMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVMapAsTable

Constructs an IVMapAsTable object.

Overload 1

```
public:IVMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMapAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMap	934	addDifference	935
addIntersection	936	addUnion	937
differenceWith	937	IAMap	934
intersectionWith	938	operator !=	938
operator ==	938	unionWith	938

IVMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2557	os_mark_schema_type	2557

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable	1391	operator =	1391

IVGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsTable	2257		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IAMap			
Definition	**Page Number**	**Definition**	**Page Number**
IAMap	934	locateNext	939
locateNextElementWithKey	939	numberOfDifferentKeys	940
numberOfElementsWithKey	940	numberOfOccurrences	940
removeAllElementsWithKey	940	removeAllOccurrences	941
setToNextWithDifferentKey	941		

IGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGMapAsTable	1391	IGMapAsTableTypedef	1391
ImplOf	1392		

IVGMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMapAsTable	2257	IVGMapAsTableTypedef	2258

Inherited Protected Data

None

IVMultiwayTree

Notification-enabled variant of IMultiwayTree.

Class header file: imwt.h

IVMultiwayTree - Hierarchy List

IMetaTree
IATreeBase
IATree
IAMultiwayTree
IGMultiwayTree
IVGMultiwayTree
IVMultiwayTree

IVMultiwayTree - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct or destruct objects of the class.

IVMultiwayTree

Constructs an IVMultiwayTree object.

Overload 1

```
public:IVMultiwayTree()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
protected:IVMultiwayTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMultiwayTree - Type Definitions

Guard

```
typedef ITreeCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVMultiwayTree < numOfChildren , Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IVMultiwayTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVMultiwayTree - Inherited Member Functions and Data

Inherited Public Functions

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
~IATree	1055	addAsChild	1055
addAsRoot	1056	adoptStreamer	1068
allElementsDo	1062	allSubtreeElementsDo	1064
childPositionAt	1067	elementAt	1066
isConsistent	1057	isEmpty	1057
isLeaf	1067	isRoot	1067
newCursor	1058	numberOfElements	1057
numberOfLeaves	1057	numberOfSubtreeElements	1058
numberOfSubtreeLeaves	1058	operator <<=	1068
operator >>=	1068	position	1068
removeAll	1056	removeSubtree	1056
replaceAt	1056	setToChild	1058
setToFirst	1059	setToFirstExistingChild	1059
setToLast	1059	setToLastExistingChild	1060
setToNext	1060	setToNextExistingChild	1060
setToParent	1061	setToPrevious	1061
setToPreviousExistingChild	1061	setToRoot	1062

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
attachAsChild	944	attachAsRoot	945
attachSubtreeAsChild	945	attachSubtreeAsRoot	946
copy	946	copySubtree	946

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
hasChild	947	numberOfChildren	947

IVMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2561	os_mark_schema_type	2561

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGMultiwayTree	1398	operator =	1398

IVGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGMultiwayTree	2261		

Inherited Public Data

None

Inherited Protected Functions

IATreeBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IATreeBase	1069	CrsrImplOf	1070
IATreeBase	1070	InterfaceFor	1070

IAMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTree	943		

IATree			
Definition	**Page Number**	**Definition**	**Page Number**
IATree	1055	ImplOf	1054

IGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IGMultiwayTree` `ImplOf`	1398 1398	`IGMultiwayTreeTypedef`	1398

IVGMultiwayTree			
Definition	**Page Number**	**Definition**	**Page Number**
`IVGMultiwayTree`	2261	`IVGMultiwayTreeTypedef`	2261

Inherited Protected Data

None

IVPriorityQueue

Variant of IPriorityQueue enabled for notifications.

Class header file: ipqu.h

IVPriorityQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueue
IVGPriorityQueue
IVPriorityQueue

IVPriorityQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVPriorityQueue

Constructs an IVPriorityQueue object.

Overload 1

```
public:IVPriorityQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVPriorityQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVPriorityQueue < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueue - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2564	os_mark_schema_type	2564

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue	1403	operator =	1403

IVGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueue	2264		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueue ImplOf	1403 1404	IGPriorityQueueTypedef	1403

IVGPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueue	2264	IVGPriorityQueueTypedef	2265

Inherited Protected Data

None

IVPriorityQueueAsDilTable

Variant of IPriorityQueueAsDilTable enabled for notifications.

Class header file: ipqudil.h

IVPriorityQueueAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsDilTable
IVGPriorityQueueAsDilTable
IVPriorityQueueAsDilTable

IVPriorityQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVPriorityQueueAsDilTable

Constructs an IVPriorityQueueAsDilTable object.

Overload 1

```
public:IVPriorityQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVPriorityQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVPriorityQueueAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2567	os_mark_schema_type	2567

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable	1407	operator =	1407

IVGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsDilTable	2268		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsDilTable	1407	IGPriorityQueueAsDilTableTypedef	1407
ImplOf	1408		

IVGPriorityQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsDilTable	2268	IVGPriorityQueueAsDilTableTypedef	2268

Inherited Protected Data

None

IVPriorityQueueAsList

Variant of IPriorityQueueAsList enabled for notifications.

Class header file: ipqulst.h

IVPriorityQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsList
IVGPriorityQueueAsList
IVPriorityQueueAsList

IVPriorityQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVPriorityQueueAsList

Constructs an IVPriorityQueueAsList object.

Overload 1

```
public:IVPriorityQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVPriorityQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVPriorityQueueAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2571	os_mark_schema_type	2571

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList	1412	operator =	1413

IVGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsList	2271		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsList ImplOf	1412 1414	IGPriorityQueueAsListTypedef	1413

IVGPriorityQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsList	2271	IVGPriorityQueueAsListTypedef	2271

Inherited Protected Data

None

IVPriorityQueueAsTable

Variant of IPriorityQueueAsTable enabled for notifications.

Class header file: ipqutab.h

IVPriorityQueueAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAPriorityQueue
IGPriorityQueueAsTable
IVGPriorityQueueAsTable
IVPriorityQueueAsTable

IVPriorityQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.
IVPriorityQueueAsTable

Constructs an IVPriorityQueueAsTable object.

Overload 1

```
public:IVPriorityQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVPriorityQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVPriorityQueueAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVPriorityQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAPriorityQueue	957	compare	958
containsAllKeysFrom	961	containsElementWithKey	961
dequeue	958	elementWithKey	962
enqueue	959	IAPriorityQueue	957

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
key	962	locateElementWithKey	959
locateNextElementWithKey	960	locateOrAddElementWithKey	960
numberOfDifferentKeys	962	numberOfElementsWithKey	962
removeFirst	961	setToNextWithDifferentKey	962

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2574	os_mark_schema_type	2574

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable	1418	operator =	1418

IVGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsTable	2274		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAPriorityQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAPriorityQueue	957		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGPriorityQueueAsTable	1418	IGPriorityQueueAsTableTypedef	1419
ImplOf	1419		

IVGPriorityQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGPriorityQueueAsTable	2274	IVGPriorityQueueAsTableTypedef	2275

Inherited Protected Data

None

IVQueue

Variant of IQueue enabled for notifications.

Class header file: iqu.h

IVQueue - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueue
IVGQueue
IVQueue

IVQueue - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVQueue

Constructs an IVQueue object.

Overload 1

```
public:IVQueue(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVQueue(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueue - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVQueue < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueue - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueue - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVQueue			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2578	os_mark_schema_type	2578

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue	1425	operator =	1426

IVGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueue	2278		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueue ImplOf	1425 1426	IGQueueTypedef	1426

IVGQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueue	2278	IVGQueueTypedef	2278

Inherited Protected Data

None

IVQueueAsDilTable

Variant of IQueueAsDilTable enabled for notifications.

Class header file: iqudil.h

IVQueueAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsDilTable
IVGQueueAsDilTable
IVQueueAsDilTable

IVQueueAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVQueueAsDilTable

Constructs an IVQueueAsDilTable object.

Overload 1

```
public:IVQueueAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVQueueAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVQueueAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2581	os_mark_schema_type	2581

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable	1429	operator =	1429

IVGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsDilTable	2281		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsDilTable ImplOf	1429 1430	IGQueueAsDilTableTypedef	1429

IVGQueueAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsDilTable	2281	IVGQueueAsDilTableTypedef	2281

Inherited Protected Data

None

IVQueueAsList

Variant of IQueueAsList enabled for notifications.

Class header file: iqulst.h

IVQueueAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAQueue
IGQueueAsList
IVGQueueAsList
IVQueueAsList

IVQueueAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVQueueAsList

Constructs an IVQueueAsList object.

Overload 1

```
public:IVQueueAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVQueueAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVQueueAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsList - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2584	os_mark_schema_type	2584

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList	1434	operator =	1435

IVGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsList	2284		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsList	1434	IGQueueAsListTypedef	1435
ImplOf	1436		

IVGQueueAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsList	2284	IVGQueueAsListTypedef	2284

Inherited Protected Data

None

IVQueueAsTable

Variant of IQueueAsTable enabled for notifications.

Class header file: iqutab.h

IVQueueAsTable - Hierarchy List

IMetaCollection
  IACollectionBase
    IARestrictedAccessCollection
      IAQueue
        IGQueueAsTable

IVQueueAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVQueueAsTable

Constructs an IVQueueAsTable object.

Overload 1

```
public:IVQueueAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVQueueAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVQueueAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVQueueAsTable - Inherited Member Functions and Data

Inherited Public Functions

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
~IAQueue	965	addAsLast	966
compare	966	dequeue	967
enqueue	967	IAQueue	965
removeFirst	968		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IVQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2587	os_mark_schema_type	2587

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	operator =	1440

IVGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsTable	2287		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IAQueue			
Definition	**Page Number**	**Definition**	**Page Number**
IAQueue	965		

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGQueueAsTable	1440	IGQueueAsTableTypedef	1440
ImplOf	1441		

IVGQueueAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGQueueAsTable	2287	IVGQueueAsTableTypedef	2288

Inherited Protected Data

None

IVRelation

Variant of IRelation enabled for notifications.

Class header file: irel.h

IVRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelation
IVGRelation
IVRelation

IVRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVRelation

Constructs an IVRelation object.

Overload 1

```
public:IVRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVRelation < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2591	os_mark_schema_type	2591

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation	1447	operator =	1447

IVGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelation	2291		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelation ImplOf	1447 1448	IGRelationTypedef	1448

IVGRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelation	2291	IVGRelationTypedef	2291

Inherited Protected Data

None

IVRelationAsDilTable

Class header file: ireldil.h

IVRelationAsDilTable - Hierarchy List

IMetaCollection

IACollectionBase

IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsDilTable
IVGRelationAsDilTable
IVRelationAsDilTable

IVRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVRelationAsDilTable

Constructs an IVRelationAsDilTable object.

Overload 1

```
public:IVRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVRelationAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
operator ==	973	unionWith	975

IVRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2594	os_mark_schema_type	2594

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable	1451	operator =	1452

IVGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsDilTable	2294		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsDilTable ImplOf	1451 1452	IGRelationAsDilTableTypedef	1452

IVGRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsDilTable	2294	IVGRelationAsDilTableTypedef	2295

Inherited Protected Data

None

IVRelationAsHshTable

Class header file: irelhsh.h

IVRelationAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsHshTable
IVGRelationAsHshTable
IVRelationAsHshTable

IVRelationAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVRelationAsHshTable

Constructs an IVRelationAsHshTable object.

Overload 1

```
public:IVRelationAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVRelationAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVRelationAsHshTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2598	os_mark_schema_type	2598

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable	1457	operator =	1457

IVGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsHshTable	2298		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsHshTable ImplOf	1457 1458	IGRelationAsHshTableTypedef	1457

IVGRelationAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsHshTable	2298	IVGRelationAsHshTableTypedef	2298

Inherited Protected Data

None

IVRelationAsList

Class header file: irellst.h

IVRelationAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsList
IVGRelationAsList
IVRelationAsList

IVRelationAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVRelationAsList

Constructs an IVRelationAsList object.

Overload 1

```
public:IVRelationAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVRelationAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVRelationAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2602	os_mark_schema_type	2602

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList	1462	operator =	1463

IVGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsList	2302		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation numberOfOccurrences	970 976	locateNext removeAllOccurrences	976 975

IGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsList ImplOf	1462 1464	IGRelationAsListTypedef	1463

IVGRelationAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsList	2302	IVGRelationAsListTypedef	2302

Inherited Protected Data

None

IVRelationAsTable

Class header file: ireltab.h

IVRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeyCollection
IARelation
IGRelationAsTable
IVGRelationAsTable
IVRelationAsTable

IVRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVRelationAsTable

Constructs an IVRelationAsTable object.

Overload 1

```
public:IVRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVRelationAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeyCollection	888	IAEqualityKeyCollection	889

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IARelation	970	addDifference	971
addIntersection	972	addUnion	973
differenceWith	974	IARelation	970
intersectionWith	974	operator !=	973
operator ==	973	unionWith	975

IVRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2606	os_mark_schema_type	2606

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable	1468	operator =	1468

IVGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsTable	2306		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeyCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeyCollection	889		

IARelation			
Definition	**Page Number**	**Definition**	**Page Number**
IARelation	970	locateNext	976
numberOfOccurrences	976	removeAllOccurrences	975

IGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGRelationAsTable	1468	IGRelationAsTableTypedef	1469
ImplOf	1469		

IVGRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGRelationAsTable	2306	IVGRelationAsTableTypedef	2306

Inherited Protected Data

None

IVSequence

Variant of ISequence enabled for notifications.

Class header file: iseq.h

IVSequence - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequence
IVGSequence
IVSequence

IVSequence - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSequence

Constructs an IVSequence object.

Overload 1

```
public:IVSequence(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSequence(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequence - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSequence < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequence - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequence - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence IASequence	992 992	compare	993

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection addAsLast addAsPrevious any first last sort	996 997 998 999 1000 1000 1001	addAsFirst addAsNext addAtPosition elementAtPosition IASequentialCollection reverse	997 998 999 1000 996 1001

IVSequence			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2610	os_mark_schema_type	2610

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence	1475	operator =	1476

IVGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequence	2309		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequence ImplOf	1475 1476	IGSequenceTypedef	1476

IVGSequence			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequence	2309	IVGSequenceTypedef	2310

Inherited Protected Data

None

IVSequenceAsDilTable

Variant of ISequenceAsDilTable enabled for notifications.

Class header file: iseqdil.h

IVSequenceAsDilTable - Hierarchy List

IMetaCollection

IACollectionBase

IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsDilTable
IVGSequenceAsDilTable
IVSequenceAsDilTable

IVSequenceAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSequenceAsDilTable

Constructs an IVSequenceAsDilTable object.

Overload 1

```
public:IVSequenceAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSequenceAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSequenceAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2613	os_mark_schema_type	2613

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable	1479	operator =	1480

IVGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsDilTable	2313		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsDilTable ImplOf	1479 1481	IGSequenceAsDilTableTypedef	1480

IVGSequenceAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsDilTable	2313	IVGSequenceAsDilTableTypedef	2314

Inherited Protected Data

None

IVSequenceAsList

Variant of ISequenceAsList enabled for notifications.

Class header file: iseqlst.h

IVSequenceAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsList

IVSequenceAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSequenceAsList

Constructs an IVSequenceAsList object.

Overload 1

```
public:IVSequenceAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSequenceAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSequenceAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2617	os_mark_schema_type	2617

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList	1485	operator =	1486

IVGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsList	2317		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsList ImplOf	1485 1487	IGSequenceAsListTypedef	1486

IVGSequenceAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsList	2317	IVGSequenceAsListTypedef	2318

Inherited Protected Data

None

IVSequenceAsTable

Variant of ISequenceAsTable enabled for notifications.

Class header file: iseqtab.h

IVSequenceAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAOrderedCollection
IASequentialCollection
IASequence
IGSequenceAsTable

IVSequenceAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSequenceAsTable

Constructs an IVSequenceAsTable object.

Overload 1

```
public:IVSequenceAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSequenceAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSequenceAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSequenceAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	anyElement	871
copy	872	disableNotification	873
elementAt	865	enableNotification	873
IACollection	863	isBounded	864
isConsistent	864	isEmpty	864
isEnabledForNotification	873	isFull	864
maxNumberOfElements	864	newCursor	866
newMngCursor	866	notifier	873
notifyObservers	874	numberOfElements	865
operator <<=	874	operator >>=	874
operator INotifier &	874	removeAll	871
removeAt	866	replaceAt	867
setToFirst	867	setToNext	867

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAOrderedCollection	949	firstElement	951
IAOrderedCollection	949	isFirst	952
isFirstAt	952	isLast	952
isLastAt	952	lastElement	952
position	953	positionAt	953
removeAtPosition	953	removeFirst	953
removeLast	954	setToLast	950
setToPosition	950	setToPrevious	950

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequence	992	compare	993
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IASequentialCollection	996	addAsFirst	997
addAsLast	997	addAsNext	998
addAsPrevious	998	addAtPosition	999
any	999	elementAtPosition	1000
first	1000	IASequentialCollection	996
last	1000	reverse	1001
sort	1001		

IVSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2621	os_mark_schema_type	2621

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable	1491	operator =	1492

IVGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsTable	2321		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAOrderedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAOrderedCollection	949		

IASequence			
Definition	**Page Number**	**Definition**	**Page Number**
IASequence	992		

IASequentialCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IASequentialCollection	996		

IGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSequenceAsTable ImplOf	1491 1493	IGSequenceAsTableTypedef	1492

IVGSequenceAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSequenceAsTable	2321	IVGSequenceAsTableTypedef	2322

Inherited Protected Data

None

IVSet

Variant of ISet enabled for notifications.

Class header file: iset.h

IVSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSet
IVGSet

IVSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSet

Constructs an IVSet object.

Overload 1

```
public:IVSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSet < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2625	os_mark_schema_type	2625

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet	1499	operator =	1500

IVGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSet	2325		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSet ImplOf	1499 1500	IGSetTypedef	1500

IVGSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSet	2325	IVGSetTypedef	2326

Inherited Protected Data

None

IVSetAsAvlTree

Variant of ISetAsAvlTree enabled for notifications.

Class header file: isetavl.h

IVSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsAvlTree
IVGSetAsAvlTree
IVSetAsAvlTree

IVSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsAvlTree

Constructs an IVSetAsAvlTree object.

Overload 1

```
public:IVSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2629	os_mark_schema_type	2629

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	operator =	1504

IVGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsAvlTree	2329		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsAvlTree	1503	IGSetAsAvlTreeTypedef	1504
ImplOf	1504		

IVGSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsAvlTree	2329	IVGSetAsAvlTreeTypedef	2329

Inherited Protected Data

None

IVSetAsBstTree

Variant of ISetAsBstTree enabled for notifications.

Class header file: isetbst.h

IVSetAsBstTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IASet
        - IGSetAsBstTree
          - IVGSetAsBstTree

IVSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsBstTree

Constructs an IVSetAsBstTree object.

Overload 1

```
public:IVSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2632	os_mark_schema_type	2632

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree	1509	operator =	1509

IVGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsBstTree	2332		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsBstTree ImplOf	1509 1510	IGSetAsBstTreeTypedef	1509

IVGSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsBstTree	2332	IVGSetAsBstTreeTypedef	2333

Inherited Protected Data

None

IVSetAsDilTable

Variant of ISetAsDilTable enabled for notifications.

Class header file: isetdil.h

IVSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsDilTable
IVGSetAsDilTable
IVSetAsDilTable

IVSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsDilTable

Constructs an IVSetAsDilTable object.

Overload 1

```
public:IVSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2636	os_mark_schema_type	2636

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	operator =	1514

IVGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsDilTable	2336		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsDilTable	1514	IGSetAsDilTableTypedef	1515
ImplOf	1515		

IVGSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsDilTable	2336	IVGSetAsDilTableTypedef	2336

Inherited Protected Data

None

IVSetAsHshTable

Variant of ISetAsHshTable enabled for notifications.

Class header file: isethsh.h

IVSetAsHshTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsHshTable
IVGSetAsHshTable

IVSetAsHshTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsHshTable

Constructs an IVSetAsHshTable object.

Overload 1

```
public:IVSetAsHshTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsHshTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsHshTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsHshTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsHshTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsHshTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2639	os_mark_schema_type	2639

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable	1519	operator =	1520

IVGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsHshTable	2339		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsHshTable ImplOf	1519 1521	IGSetAsHshTableTypedef	1520

IVGSetAsHshTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsHshTable	2339	IVGSetAsHshTableTypedef	2340

Inherited Protected Data

None

IVSetAsList

Variant of ISetAsList enabled for notifications.

Class header file: isetlst.h

IVSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IASet
IGSetAsList
IVGSetAsList
IVSetAsList

IVSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsList

Constructs an IVSetAsList object.

Overload 1

```
public:IVSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2643	os_mark_schema_type	2643

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	operator =	1525

IVGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsList	2343		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsList	1525	IGSetAsListTypedef	1525
ImplOf	1526		

IVGSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsList	2343	IVGSetAsListTypedef	2343

Inherited Protected Data

None

IVSetAsTable

Variant of ISetAsTable enabled for notifications.

Class header file: isettab.h

IVSetAsTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IASet
        - IGSetAsTable
          - IVGSetAsTable

IVSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSetAsTable

Constructs an IVSetAsTable object.

Overload 1

```
public:IVSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSetAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateOrAdd	885
remove	886	removeAllOccurrences	887

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASet	1003	addDifference	1004
addIntersection	1005	addUnion	1006
differenceWith	1006	IASet	1003
intersectionWith	1007	operator !=	1007
operator ==	1007	unionWith	1008

IVSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2646	os_mark_schema_type	2646

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable	1530	operator =	1531

IVGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsTable	2346		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IASet			
Definition	**Page Number**	**Definition**	**Page Number**
IASet	1003	locateNext	1008
numberOfOccurrences	1009	removeAllOccurrences	1009

IGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSetAsTable ImplOf	1530 1531	IGSetAsTableTypedef	1531

IVGSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSetAsTable	2346	IVGSetAsTableTypedef	2347

Inherited Protected Data

None

IVSortedBag

Variant of ISortedBag enabled for notifications.

Class header file: isb.h

IVSortedBag - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedBag
          - IGSortedBag
            - IVGSortedBag
              - **IVSortedBag**

IVSortedBag - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBag

Constructs an IVSortedBag object.

Overload 1

```
public:IVSortedBag(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBag(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBag - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBag < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBag - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBag - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2650	os_mark_schema_type	2650

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag	1537	operator =	1537

IVGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBag	2350		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBag ImplOf	1537 1538	IGSortedBagTypedef	1538

IVGSortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBag	2350	IVGSortedBagTypedef	2350

Inherited Protected Data

None

IVSortedBagAsAvlTree

Variant of ISortedBagAsAvlTree enabled for notifications.

Class header file: isbavl.h

IVSortedBagAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsAvlTree
IVGSortedBagAsAvlTree
IVSortedBagAsAvlTree

IVSortedBagAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBagAsAvlTree

Constructs an IVSortedBagAsAvlTree object.

Overload 1

```
public:IVSortedBagAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBagAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBagAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2654	os_mark_schema_type	2654

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree	1541	operator =	1542

IVGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsAvlTree	2354		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsAvlTree ImplOf	1541 1543	IGSortedBagAsAvlTreeTypedef	1542

IVGSortedBagAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsAvlTree	2354	IVGSortedBagAsAvlTreeTypedef	2354

Inherited Protected Data

None

IVSortedBagAsBstTree

Variant of ISortedBagAsBstTree enabled for notifications.

Class header file: isbbst.h

IVSortedBagAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsBstTree
IVGSortedBagAsBstTree
IVSortedBagAsBstTree

IVSortedBagAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBagAsBstTree

Constructs an IVSortedBagAsBstTree object.

Overload 1

```
public:IVSortedBagAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBagAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBagAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2658	os_mark_schema_type	2658

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree	1547	operator =	1548

IVGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsBstTree	2358		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsBstTree ImplOf	1547 1549	IGSortedBagAsBstTreeTypedef	1548

IVGSortedBagAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsBstTree	2358	IVGSortedBagAsBstTreeTypedef	2358

Inherited Protected Data

None

IVSortedBagAsDilTable

Variant of ISortedBagAsDilTable enabled for notifications.

Class header file: isbdil.h

IVSortedBagAsDilTable - Hierarchy List

IMetaCollection

IACollectionBase

IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsDilTable
IVGSortedBagAsDilTable
IVSortedBagAsDilTable

IVSortedBagAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBagAsDilTable

Constructs an IVSortedBagAsDilTable object.

Overload 1

```
public:IVSortedBagAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBagAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBagAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016
unionWith	1017		

IVSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2661	os_mark_schema_type	2661

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable	1554	operator =	1554

IVGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsDilTable	2362		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsDilTable ImplOf	1554 1555	IGSortedBagAsDilTableTypedef	1554

IVGSortedBagAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsDilTable	2362	IVGSortedBagAsDilTableTypedef	2362

Inherited Protected Data

None

IVSortedBagAsList

Variant of ISortedBagAsList enabled for notifications.

Class header file: isblst.h

IVSortedBagAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsList

IVSortedBagAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBagAsList

Constructs an IVSortedBagAsList object.

Overload 1

```
public:IVSortedBagAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBagAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBagAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag numberOfDifferentElements operator == unionWith	1011 1016 1016 1017	intersectionWith operator != setToNextDifferentElement	1015 1016 1016

IVSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2665	os_mark_schema_type	2665

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList	1560	operator =	1560

IVGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsList	2365		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsList ImplOf	1560 1561	IGSortedBagAsListTypedef	1560

IVGSortedBagAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsList	2365	IVGSortedBagAsListTypedef	2366

Inherited Protected Data

None

IVSortedBagAsTable

Variant of ISortedBagAsTable enabled for notifications.

Class header file: isbtab.h

IVSortedBagAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedBag
IGSortedBagAsTable
IVGSortedBagAsTable
IVSortedBagAsTable

IVSortedBagAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedBagAsTable

Constructs an IVSortedBagAsTable object.

Overload 1

```
public:IVSortedBagAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedBagAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedBagAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedBagAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	numberOfOccurrences	886
remove	886	removeAllOccurrences	887

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedBag	1011	addDifference	1013
addIntersection	1014	addUnion	1014
compare	1012	differenceWith	1015
IASortedBag	1011	intersectionWith	1015
numberOfDifferentElements	1016	operator !=	1016
operator ==	1016	setToNextDifferentElement	1016

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
unionWith	1017		

IVSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2669	os_mark_schema_type	2669

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable	1566	operator =	1566

IVGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsTable	2369		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedBag			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedBag	1011		

IGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedBagAsTable ImplOf	1566 1567	IGSortedBagAsTableTypedef	1567

IVGSortedBagAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedBagAsTable	2369	IVGSortedBagAsTableTypedef	2370

Inherited Protected Data

None

IVSortedMap

Variant of ISortedMap enabled for notifications.

Class header file: ism.h

IVSortedMap - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - IGSortedMap
            - IVGSortedMap
              - **IVSortedMap**

IVSortedMap - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMap

Constructs an IVSortedMap object.

Overload 1

```
public:IVSortedMap(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMap(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMap - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMap < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IVSortedMap - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMap - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

IVSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2673	os_mark_schema_type	2673

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	operator =	1574

IVGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMap	2373		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMap	1574	IGSortedMapTypedef	1574
ImplOf	1575		

IVGSortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMap	2373	IVGSortedMapTypedef	2373

Inherited Protected Data

None

IVSortedMapAsAvlTree

Variant of ISortedMapAsAvlTree enabled for notifications.

Class header file: ismavl.h

IVSortedMapAsAvlTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - IGSortedMapAsAvlTree
            - IVGSortedMapAsAvlTree
              - **IVSortedMapAsAvlTree**

IVSortedMapAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMapAsAvlTree

Constructs an IVSortedMapAsAvlTree object.

Overload 1

```
public:IVSortedMapAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMapAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMapAsAvlTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

IVSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2677	os_mark_schema_type	2677

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	operator =	1579

IVGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsAvlTree	2377		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsAvlTree	1578	IGSortedMapAsAvlTreeTypedef	1579
ImplOf	1580		

IVGSortedMapAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsAvlTree	2377	IVGSortedMapAsAvlTreeTypedef	2377

Inherited Protected Data

None

IVSortedMapAsBstTree

Variant of ISortedMapAsBstTree enabled for notifications.

Class header file: ismbst.h

IVSortedMapAsBstTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsBstTree
IVGSortedMapAsBstTree
IVSortedMapAsBstTree

IVSortedMapAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMapAsBstTree

Constructs an IVSortedMapAsBstTree object.

Overload 1

```
public:IVSortedMapAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMapAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMapAsBstTree < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap operator != unionWith	1023 1027 1028	intersectionWith operator ==	1027 1027

IVSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2681	os_mark_schema_type	2681

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	operator =	1585

IVGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsBstTree	2381		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsBstTree	1584	IGSortedMapAsBstTreeTypedef	1585
ImplOf	1586		

IVGSortedMapAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsBstTree	2381	IVGSortedMapAsBstTreeTypedef	2381

Inherited Protected Data

None

IVSortedMapAsDilTable

Class header file: ismdil.h

IVSortedMapAsDilTable - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - IGSortedMapAsDilTable
            - IVGSortedMapAsDilTable
              - **IVSortedMapAsDilTable**

IVSortedMapAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMapAsDilTable

Constructs an IVSortedMapAsDilTable object.

Overload 1

```
public:IVSortedMapAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMapAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMapAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2685	os_mark_schema_type	2685

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	operator =	1591

IVGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsDilTable	2385		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsDilTable	1591	IGSortedMapAsDilTableTypedef	1591
ImplOf	1592		

IVGSortedMapAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsDilTable	2385	IVGSortedMapAsDilTableTypedef	2385

Inherited Protected Data

None

IVSortedMapAsList

Variant of ISortedMapAsList enabled for notifications.

Class header file: ismlst.h

IVSortedMapAsList - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualityKeySortedCollection
        - IASortedMap
          - IGSortedMapAsList
            - IVGSortedMapAsList
              - **IVSortedMapAsList**

IVSortedMapAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMapAsList

Constructs an IVSortedMapAsList object.

Overload 1

```
public:IVSortedMapAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMapAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMapAsList < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2689	os_mark_schema_type	2689

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	operator =	1597

IVGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsList	2389		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap	1023	numberOfDifferentKeys	1028
numberOfElementsWithKey	1028	numberOfOccurrences	1028
removeAllElementsWithKey	1029	removeAllOccurrences	1029
setToNextWithDifferentKey	1029		

IGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsList	1597	IGSortedMapAsListTypedef	1597
ImplOf	1598		

IVGSortedMapAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsList	2389	IVGSortedMapAsListTypedef	2389

Inherited Protected Data

None

IVSortedMapAsTable

Variant of ISortedMapAsTable enabled for notifications.

Class header file: ismtab.h

IVSortedMapAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedMap
IGSortedMapAsTable
IVGSortedMapAsTable
IVSortedMapAsTable

IVSortedMapAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedMapAsTable

Constructs an ISortedMapAsTable object.

Overload 1

```
public:IVSortedMapAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedMapAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedMapAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedMapAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedMap	1023	addDifference	1025
addIntersection	1025	addUnion	1026
compare	1024	differenceWith	1027
IASortedMap	1023	intersectionWith	1027
operator !=	1027	operator ==	1027
unionWith	1028		

IVSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2693	os_mark_schema_type	2693

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsTable	1603	operator =	1603

IVGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsTable	2392		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedMap numberOfElementsWithKey removeAllElementsWithKey	1023 1028 1029	numberOfDifferentKeys numberOfOccurrences removeAllOccurrences	1028 1028 1029

IASortedMap			
Definition	**Page Number**	**Definition**	**Page Number**
setToNextWithDifferentKey	1029		

IGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedMapAsTable ImplOf	1603 1604	IGSortedMapAsTableTypedef	1604

IVGSortedMapAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedMapAsTable	2392	IVGSortedMapAsTableTypedef	2393

Inherited Protected Data

None

IVSortedRelation

Variant of ISortedRelation enabled for notifications.

Class header file: isr.h

IVSortedRelation - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelation
IVGSortedRelation
IVSortedRelation

IVSortedRelation - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedRelation

Constructs an IVSortedRelation object.

Overload 1

```
public:IVSortedRelation(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedRelation(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelation - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedRelation < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelation - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelation - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2697	os_mark_schema_type	2697

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation	1611	operator =	1611

IVGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelation	2396		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelation ImplOf	1611 1612	IGSortedRelationTypedef	1612

IVGSortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelation	2396	IVGSortedRelationTypedef	2397

Inherited Protected Data

None

IVSortedRelationAsDilTable

Variant of ISortedRelationAsDilTable enabled for notifications.

Class header file: isrdil.h

IVSortedRelationAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsDilTable
IVGSortedRelationAsDilTable
IVSortedRelationAsDilTable

IVSortedRelationAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedRelationAsDilTable

Constructs an IVSortedRelationAsDilTable object.

Overload 1

```
public:IVSortedRelationAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedRelationAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedRelationAsDilTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2701	os_mark_schema_type	2701

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable	1615	operator =	1616

IVGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelationAsDilTable	2400		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsDilTable ImplOf	1615 1617	IGSortedRelationAsDilTableTypedef	1616

IVGSortedRelationAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelationAsDilTable	2400	IVGSortedRelationAsDilTableTypedef	2400

Inherited Protected Data

None

IVSortedRelationAsTable

Variant of ISortedRelationAsTable enabled for notifications.

Class header file: isrtab.h

IVSortedRelationAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualityKeySortedCollection
IASortedRelation
IGSortedRelationAsTable
IVGSortedRelationAsTable
IVSortedRelationAsTable

IVSortedRelationAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedRelationAsTable

Constructs an IVSortedRelationAsTable object.

Overload 1

```
public:IVSortedRelationAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedRelationAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedRelationAsTable < Element , Key > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedRelationAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityKeySortedCollection	892	IAEqualityKeySortedCollection	892

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedRelation	1032	addDifference	1034
addIntersection	1034	addUnion	1035
compare	1033	differenceWith	1036
IASortedRelation	1032	intersectionWith	1036
operator !=	1036	operator ==	1037
unionWith	1037		

IVSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2705	os_mark_schema_type	2705

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable	1621	operator =	1622

IVGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelationAsTable	2404		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualityKeySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityKeySortedCollection	892		

IASortedRelation			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedRelation removeAllOccurrences	1032 1038	numberOfOccurrences	1038

IGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedRelationAsTable ImplOf	1621 1623	IGSortedRelationAsTableTypedef	1622

IVGSortedRelationAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedRelationAsTable	2404	IVGSortedRelationAsTableTypedef	2404

Inherited Protected Data

None

IVSortedSet

Variant of ISortedSet enabled for notifications.

Class header file: iss.h

IVSortedSet - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSet
IVGSortedSet
IVSortedSet

IVSortedSet - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSet

Constructs an IVSortedSet object.

Overload 1

```
public:IVSortedSet(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSet(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSet - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSet < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This member is not accessible.

IVSortedSet - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSet - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2709	os_mark_schema_type	2709

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet	1629	operator =	1630

IVGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSet	2408		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSet ImplOf	1629 1630	IGSortedSetTypedef	1629

IVGSortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSet	2408	IVGSortedSetTypedef	2408

Inherited Protected Data

None

IVSortedSetAsAvlTree

Variant of ISortedSetAsAvlTree enabled for notifications.

Class header file: issavl.h

IVSortedSetAsAvlTree - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsAvlTree
IVGSortedSetAsAvlTree
IVSortedSetAsAvlTree

IVSortedSetAsAvlTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSetAsAvlTree

Constructs an IVSortedSetAsAvlTree object.

Overload 1

```
public:IVSortedSetAsAvlTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSetAsAvlTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsAvlTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSetAsAvlTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsAvlTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsAvlTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2712	os_mark_schema_type	2712

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree	1633	operator =	1634

IVGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsAvlTree	2412		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsAvlTree ImplOf	1633 1635	IGSortedSetAsAvlTreeTypedef	1634

IVGSortedSetAsAvlTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsAvlTree	2412	IVGSortedSetAsAvlTreeTypedef	2412

Inherited Protected Data

None

IVSortedSetAsBstTree

Variant of ISortedSetAsBstTree enabled for notifications.

Class header file: issbst.h

IVSortedSetAsBstTree - Hierarchy List

IMetaCollection
- IACollectionBase
  - IACollection
    - IAEqualityCollection
      - IAEqualitySortedCollection
        - IASortedSet

IGSortedSetAsBstTree
IVGSortedSetAsBstTree
IVSortedSetAsBstTree

IVSortedSetAsBstTree - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSetAsBstTree

Constructs an IVSortedSetAsBstTree object.

Overload 1

```
public:IVSortedSetAsBstTree(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSetAsBstTree(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsBstTree - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSetAsBstTree < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsBstTree - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsBstTree - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet operator != unionWith	1041 1045 1045	intersectionWith operator ==	1045 1045

IVSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2716	os_mark_schema_type	2716

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree	1639	operator =	1640

IVGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsBstTree	2416		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsBstTree ImplOf	1639 1641	IGSortedSetAsBstTreeTypedef	1640

IVGSortedSetAsBstTree			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsBstTree	2416	IVGSortedSetAsBstTreeTypedef	2416

Inherited Protected Data

None

IVSortedSetAsDilTable

Variant of ISortedSetAsDilTable enabled for notifications.

Class header file: issdil.h

IVSortedSetAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsDilTable
IVGSortedSetAsDilTable
IVSortedSetAsDilTable

IVSortedSetAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSetAsDilTable

Constructs an IVSortedSetAsDilTable object.

Overload 1

```
public:IVSortedSetAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSetAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsDilTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSetAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2720	os_mark_schema_type	2720

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable	1645	operator =	1646

IVGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsDilTable	2419		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsDilTable ImplOf	1645 1647	IGSortedSetAsDilTableTypedef	1646

IVGSortedSetAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsDilTable	2419	IVGSortedSetAsDilTableTypedef	2420

Inherited Protected Data

None

IVSortedSetAsList

Variant of ISortedSetAsList enabled for notifications.

Class header file: isslst.h

IVSortedSetAsList - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsList
IVGSortedSetAsList
IVSortedSetAsList

IVSortedSetAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSetAsList

Constructs an IVSortedSetAsList object.

Overload 1

```
public:IVSortedSetAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSetAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsList - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSetAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsList - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2724	os_mark_schema_type	2724

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList	1652	operator =	1652

IVGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsList	2423		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet	1041	numberOfOccurrences	1046
removeAllOccurrences	1046		

IGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsList ImplOf	1652 1653	IGSortedSetAsListTypedef	1652

IVGSortedSetAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsList	2423	IVGSortedSetAsListTypedef	2424

Inherited Protected Data

None

IVSortedSetAsTable

Variant of ISortedSetAsTable enabled for notifications.

Class header file: isstab.h

IVSortedSetAsTable - Hierarchy List

IMetaCollection
IACollectionBase
IACollection
IAEqualityCollection
IAEqualitySortedCollection
IASortedSet
IGSortedSetAsTable
IVGSortedSetAsTable
IVSortedSetAsTable

IVSortedSetAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVSortedSetAsTable

Constructs an IVSortedSetAsTable object.

Overload 1

```
public:IVSortedSetAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVSortedSetAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsTable - Type Definitions

Guard

```
typedef ICollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVSortedSetAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVSortedSetAsTable - Inherited Member Functions and Data

Inherited Public Functions

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollection	863	add	865
addAllFrom	870	adoptStreamer	874
allElementsDo	868	any	871
anyElement	871	copy	872
disableNotification	873	elementAt	865
enableNotification	873	IACollection	863
isBounded	864	isConsistent	864
isEmpty	864	isEnabledForNotification	873
isFull	864	maxNumberOfElements	864
newCursor	866	newMngCursor	866
notifier	873	notifyObservers	874
numberOfElements	865	operator <<=	874
operator >>=	874	operator INotifier &	874
removeAll	871	removeAt	866
replaceAt	867	setToFirst	867

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
setToNext	867		

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualityCollection	883	contains	884
containsAllFrom	884	IAEqualityCollection	883
locate	884	locateNext	885
locateOrAdd	885	remove	886
removeAllOccurrences	887		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IAEqualitySortedCollection	900	IAEqualitySortedCollection	900

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
~IASortedSet	1041	addDifference	1042
addIntersection	1043	addUnion	1044
compare	1042	differenceWith	1044
IASortedSet	1041	intersectionWith	1045
operator !=	1045	operator ==	1045
unionWith	1045		

IVSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2728	os_mark_schema_type	2728

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable	1658	operator =	1658

IVGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsTable	2427		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IACollection			
Definition	**Page Number**	**Definition**	**Page Number**
IACollection	863	ImplOf	873

IAEqualityCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualityCollection	883		

IAEqualitySortedCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IAEqualitySortedCollection	900		

IASortedSet			
Definition	**Page Number**	**Definition**	**Page Number**
IASortedSet removeAllOccurrences	1041 1046	numberOfOccurrences	1046

IGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGSortedSetAsTable ImplOf	1658 1659	IGSortedSetAsTableTypedef	1658

IVGSortedSetAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGSortedSetAsTable	2427	IVGSortedSetAsTableTypedef	2427

Inherited Protected Data

None

IVStack

Variant of IStack enabled for notifications.

Class header file: istk.h

IVStack - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStack
IVGStack
IVStack

IVStack - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVStack

Constructs an IVStack object.

Overload 1

```
public:IVStack(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVStack(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStack - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVStack < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStack - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStack - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVStack			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2732	os_mark_schema_type	2732

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IGStack	1665	operator =	1666

IVGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStack	2431		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IGStack ImplOf	1665 1667	IGStackTypedef	1666

IVGStack			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStack	2431	IVGStackTypedef	2431

Inherited Protected Data

None

IVStackAsDilTable

Variant of IStackAsDilTable enabled for notifications.

Class header file: istkdil.h

IVStackAsDilTable - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsDilTable
IVGStackAsDilTable
IVStackAsDilTable

IVStackAsDilTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVStackAsDilTable

Constructs an IVStackAsDilTable object.

Overload 1

```
public:IVStackAsDilTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVStackAsDilTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsDilTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVStackAsDilTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsDilTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsDilTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2735	os_mark_schema_type	2735

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable	1669	operator =	1669

IVGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsDilTable	2434		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsDilTable ImplOf	1669 1670	IGStackAsDilTableTypedef	1670

IVGStackAsDilTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsDilTable	2434	IVGStackAsDilTableTypedef	2435

Inherited Protected Data

None

IVStackAsList

Variant of IStackAsList enabled for notifications.

Class header file: istklst.h

IVStackAsList - Hierarchy List

IMetaCollection
IACollectionBase
IARestrictedAccessCollection
IAStack
IGStackAsList
IVGStackAsList
IVStackAsList

IVStackAsList - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVStackAsList

Constructs an IVStackAsList object.

Overload 1

```
public:IVStackAsList(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVStackAsList(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsList - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVStackAsList < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsList - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsList - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2738	os_mark_schema_type	2738

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	operator =	1675

IVGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsList	2437		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase	875	CrsrImplOf	876
IACollectionBase	875	InterfaceFor	876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsList	1674	IGStackAsListTypedef	1675
ImplOf	1676		

IVGStackAsList			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsList	2437	IVGStackAsListTypedef	2438

Inherited Protected Data

None

IVStackAsTable

Variant of IStackAsTable enabled for notifications.

Class header file: istktab.h

IVStackAsTable - Hierarchy List

IMetaCollection

IACollectionBase

IARestrictedAccessCollection

IAStack
IGStackAsTable
IVGStackAsTable
IVStackAsTable

IVStackAsTable - Member Functions and Data by Group

Constructors & Destructor

These member functions construct and destruct objects of this class.

IVStackAsTable

Constructs an IVGDeque object.

Overload 1

```
public:IVStackAsTable(INumber = 100)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a collection. INumber is the estimated maximum number of elements contained in the collection. The collection is unbounded and is initially empty. If the estimated maximum is exceeded, the collection is automatically enlarged.

Overload 2

```
protected:IVStackAsTable(PureImplementation&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsTable - Type Definitions

Guard

```
typedef IRestrictedAccessCollectionGuard < Element > Guard
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

PureImplementation

```
typedef TYPENAME Inherited::PureImplementation PureImplementation
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Self

```
typedef IVStackAsTable < Element > Self
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsTable - Associated Globals

get_os_typespec

```
static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IVStackAsTable - Inherited Member Functions and Data

Inherited Public Functions

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
~IARestrictedAccessCollection	978	add	979
addAllFrom	980	adoptStreamer	990
allElementsDo	985	any	986
anyElement	986	copy	982
disableNotification	989	elementAt	987
elementAtPosition	987	enableNotification	989
first	988	firstElement	988
IARestrictedAccessCollection	978	isBounded	981
isConsistent	981	isEmpty	982
isEnabledForNotification	989	isFirst	987
isFirstAt	987	isFull	982
isLast	988	isLastAt	988
last	988	lastElement	989
maxNumberOfElements	982	newCursor	983
newMngCursor	983	notifier	989
notifyObservers	989	numberOfElements	982
operator <<=	990	operator >>=	990
operator INotifier &	990	position	983
positionAt	983	removeAll	981
setToFirst	984	setToLast	984
setToNext	984	setToPosition	985
setToPrevious	985		

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
~IAStack	1049	addAsLast	1051
compare	1050	IAStack	1049
pop	1051	push	1052
removeLast	1052	top	1052

IVStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
get_os_typespec	2741	os_mark_schema_type	2741

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable	1680	operator =	1680

IVGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsTable	2441		

Inherited Public Data

None

Inherited Protected Functions

IACollectionBase			
Definition	**Page Number**	**Definition**	**Page Number**
~IACollectionBase IACollectionBase	875 875	CrsrImplOf InterfaceFor	876 876

IARestrictedAccessCollection			
Definition	**Page Number**	**Definition**	**Page Number**
IARestrictedAccessCollection	978	ImplOf	978

IAStack			
Definition	**Page Number**	**Definition**	**Page Number**
IAStack	1049		

IGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IGStackAsTable ImplOf	1680 1681	IGStackAsTableTypedef	1680

IVGStackAsTable			
Definition	**Page Number**	**Definition**	**Page Number**
IVGStackAsTable	2441	IVGStackAsTableTypedef	2441

Inherited Protected Data

None

Chapter 7. Exceptions Signaled By Collection Classes

IChildAlreadyExistsException

The IChildAlreadyExistsException class represents an exception. It is throw when you try to add a child to a tree (using the addAsChild member function) at a position that already contains a child.

Class header file: iexc.h

IChildAlreadyExistsException - Hierarchy List

IException

IPreconditionViolation

IChildAlreadyExistsException

IChildAlreadyExistsException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IChildAlreadyExistsException object from another.

~IChildAlreadyExistsException

```
public:virtual ~IChildAlreadyExistsException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IChildAlreadyExistsException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IChildAlreadyExistsException( char const*,
                                     ioc::unumeric_t = 0,
                                     Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

© Copyright IBM Corp. 2001

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IChildAlreadyExistsException( IChildAlreadyExistsException const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IChildAlreadyExistsException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768

IException			
Definition	**Page Number**	**Definition**	**Page Number**
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

ICollectionLockException

The ICollectionLockException class represents an exception. It is throw when an internal lock request fails.

Class header file: iexc.h

ICollectionLockException - Hierarchy List

IException

ICollectionLockException

ICollectionLockException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one ICollectionLockException object from another.

~ICollectionLockException

```
public:virtual ~ICollectionLockException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICollectionLockException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICollectionLockException( char const*,
                                 ioc::unumeric_t = 0,
                                 Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:ICollectionLockException(ICollectionLockException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

ICollectionLockException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

ICollectionLockTimeOutException

The ICollectionLockTimeOutException class represents an exception. It is throw when a timeout value is specified, during the construction of a Guard object, and the lock cannoat be obtained within the specified period of time.

Class header file: iexc.h

ICollectionLockTimeOutException - Hierarchy List

IException

ICollectionLockTimeOutException

ICollectionLockTimeOutException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one ICollectionLockTimeOutException object from another.

~ICollectionLockTimeOutException

```
public:virtual ~ICollectionLockTimeOutException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICollectionLockTimeOutException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICollectionLockTimeOutException( char const*,
                                        ioc::unumeric_t = 0,
                                        Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:ICollectionLockTimeOutException( ICollectionLockTimeOutException
  const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

ICollectionLockTimeOutException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

ICollectionResourceException

The ICollectionResourceException class represents an exception. It is throw during the construction of a collection when the creation of the internal resource object fails.

Class header file: iexc.h

ICollectionResourceException - Hierarchy List

IException

ICollectionResourceException

ICollectionResourceException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one ICollectionResourceException object from another.

~ICollectionResourceException

```
public:virtual ~ICollectionResourceException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICollectionResourceException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICollectionResourceException( char const*,
                                     ioc::unumeric_t = 0,
                                     Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:ICollectionResourceException( ICollectionResourceException const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

ICollectionResourceException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

ICollectionUnlockException

The ICollectionUnlockException class represents an exception. It is throw when an internal unlock request fails.

Class header file: iexc.h

ICollectionUnlockException - Hierarchy List

IException

ICollectionUnlockException

ICollectionUnlockException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one ICollectionUnlockException object from another.

~ICollectionUnlockException

```
public:virtual ~ICollectionUnlockException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ICollectionUnlockException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:ICollectionUnlockException( char const*,
                                   ioc::unumeric_t = 0,
                                   Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:ICollectionUnlockException( ICollectionUnlockException const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

ICollectionUnlockException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IEmptyException

The IEmptyException class represents an exception. It is thrown when a function tries to access an element of an empty collection. Functions that might cause this exception include firstElement and removeFirstElement.

Class header file: iexc.h

IEmptyException - Hierarchy List

IException

IPreconditionViolation

IEmptyException

IEmptyException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IEmptyException object from another.

~IEmptyException

```
public:virtual ~IEmptyException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IEmptyException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IEmptyException( char const*,
                        ioc::unumeric_t = 0,
                        Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IEmptyException(IEmptyException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IEmptyException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770

IException			
Definition	**Page Number**	**Definition**	**Page Number**
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IFullException

The IFullException class represents an exception. It is thrown when a function tries to add an element to a bounded collection that is already full. Functions that might cause this exception include add and addAsFirst.

Class header file: iexc.h

IFullException - Hierarchy List

IException

IPreconditionViolation

IFullException

IFullException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IFullException object from another.

~IFullException

```
public:virtual ~IFullException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IFullException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IFullException( char const*,
                       ioc::unumeric_t = 0,
                       Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IFullException(IFullException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IFullException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IIdenticalCollectionException

The IIdenticalCollectionException class represents an exception. It is thrown when the function addAllFrom is called with the source collection being the same as the target collection.

Class header file: iexc.h

IIdenticalCollectionException - Hierarchy List

IException

IPreconditionViolation

IIdenticalCollectionException

IIdenticalCollectionException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IIdenticalCollectionException object from another.

~IIdenticalCollectionException

```
public:virtual ~IIdenticalCollectionException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IIdenticalCollectionException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IIdenticalCollectionException( char const*,
                                      ioc::unumeric_t = 0,
                                      Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IIdenticalCollectionException( IIdenticalCollectionException const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IIdenticalCollectionException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768

IException			
Definition	**Page Number**	**Definition**	**Page Number**
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

INotBoundedException

The INotBoundedException class represents an exception. It is thrown when the function maxNumberOfElements is applied to a collection that is not bounded.

Class header file: iexc.h

INotBoundedException - Hierarchy List

IException

IPreconditionViolation

INotBoundedException

INotBoundedException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IChildAlreadyExistsException object from another.

~INotBoundedException

```
public:virtual ~INotBoundedException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

INotBoundedException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:INotBoundedException( char const*,
                             ioc::unumeric_t = 0,
                             Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:INotBoundedException(INotBoundedException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

INotBoundedException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

INotContainsKeyException

The INotContainsKeyException class represents an exception. It is thrown when the function elementWIthKey is applied to a collection that does not contain an element with the specified key.

Class header file: iexc.h

INotContainsKeyException - Hierarchy List

IException

IPreconditionViolation

INotContainsKeyException

INotContainsKeyException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IChildAlreadyExistsException object from another.

~INotContainsKeyException

```
public:virtual ~INotContainsKeyException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

INotContainsKeyException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:INotContainsKeyException( char const*,
                                 ioc::unumeric_t = 0,
                                 Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText

The text describing this particular error.

errorId

The identifier you want to associate with this particular error.

severity

Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:INotContainsKeyException(INotContainsKeyException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

INotContainsKeyException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768

IException			
Definition	**Page Number**	**Definition**	**Page Number**
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IPositionInvalidException

The IPositionInvalidException class represents an exception. It is thrown when a function specifies a position that is not valid in a collection. The functions that might cause this exception include elementAtPosition, removeAtPosition, and setToPosition.

Class header file: iexc.h

IPositionInvalidException - Hierarchy List

IException

IPreconditionViolation

IPositionInvalidException

IPositionInvalidException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IPositionInvalidException object from another.

~IPositionInvalidException

```
public:virtual ~IPositionInvalidException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IPositionInvalidException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IPositionInvalidException( char const*,
                                  ioc::unumeric_t = 0,
                                  Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IPositionInvalidException(IPositionInvalidException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IPositionInvalidException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
`baseLibrary`	766	`CLibrary`	766
`operatingSystem`	766	`other`	767
`presentationSystem`	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IPreconditionViolation

The IPreconditionViolation class represents an exception. A **precondition** of a called function is a condition that the function requires to be true when it is called. The calling function must assure that this condition holds.

A call to a function that violates the function's preconditions has two possible results:

- If the called function checks its preconditions, the function will throw an exception.
- If the function does not check its preconditions, the behaviour of the function is undefined.

Class header file: iexc.h

IPreconditionViolation - Hierarchy List

IException

IPreconditionViolation

INotBoundedException

IEmptyException

IRootAlreadyExistsException

INoConversionFunctionDefined

IContainerObjectIsNotReadYet

IFullException

IUnlockInvalidException

IPositionInvalidException

IContainedStreamedBeforeEnclosing

IIdenticalCollectionException

INotEmptyException

IChildAlreadyExistsException

INoRestoreFunctionDefined

INotContainsKeyException

IPreconditionViolation - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IPreconditionViolation object from another.

~IPreconditionViolation

```
public:virtual ~IPreconditionViolation()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IPreconditionViolation

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IPreconditionViolation( char const*,
                               ioc::unumeric_t = 0,
                               Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IPreconditionViolation(IPreconditionViolation const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IPreconditionViolation - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IRootAlreadyExistsException

The IRootAlreadyExistsException class represents an exception. It is thrown when the function addAsRoot is called for a tree that already has a root.

Class header file: iexc.h

IRootAlreadyExistsException - Hierarchy List

IException

IPreconditionViolation

IRootAlreadyExistsException

IRootAlreadyExistsException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IRootAlreadyExistsException object from another.

~IRootAlreadyExistsException

```
public:virtual ~IRootAlreadyExistsException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IRootAlreadyExistsException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IRootAlreadyExistsException( char const*,
                                    ioc::unumeric_t = 0,
                                    Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IRootAlreadyExistsException( IRootAlreadyExistsException const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IRootAlreadyExistsException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769

IException			
Definition	**Page Number**	**Definition**	**Page Number**
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

IPreconditionViolation			
Definition	**Page Number**	**Definition**	**Page Number**
~IPreconditionViolation	2773	IPreconditionViolation	2773

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamerInvalidException

The IStreamerInvalidException class represents an exception.

Class header file: iexc.h

IStreamerInvalidException - Hierarchy List

IException

IStreamerInvalidException

IStreamerInvalidException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IStreamerInvalidException object from another.

~IStreamerInvalidException

```
public:virtual ~IStreamerInvalidException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStreamerInvalidException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IStreamerInvalidException( char const*,
                                  ioc::unumeric_t = 0,
                                  Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IStreamerInvalidException(IStreamerInvalidException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IStreamerInvalidException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IStreamErrorException

The IStreamErrorException class represents an exception.

Class header file: iexc.h

IStreamErrorException - Hierarchy List

IException

IStreamErrorException

IStreamErrorException - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IStreamErrorException object from another.

~IStreamErrorException

```
public:virtual ~IStreamErrorException()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IStreamErrorException

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IStreamErrorException( char const*,
                              ioc::unumeric_t = 0,
                              Severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IStreamErrorException(IStreamErrorException const&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual char const* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

Throw Support

This member function is used to throw the exception.

throwIt

```
public:static void
  throwIt( ioc::unumeric_t,
           char const*,
           ioc::unumeric_t )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Uses the ITHROW macro to throw the exception object. The first parameter represents the message identifier of the message to insert in the exception. The second and third parameters represent the file name and line number respectively of the file where the exception occurred.

IStreamErrorException - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
name	770	nameW	770
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766

IException			
Definition	**Page Number**	**Definition**	**Page Number**
`operatingSystem` `presentationSystem`	766 767	`other`	767

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 8. Operations For Collection Classes

IAMultiwayTreeOps

Class header file: iimwt.h

IAMultiwayTreeOps - Hierarchy List

IAMultiwayTreeOps

IMultiwayTreeOps

IAMultiwayTreeOps - Member Functions and Data by Group

Miscellaneous Members

~IAMultiwayTreeOps

```
public:virtual ~IAMultiwayTreeOps()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign

```
public:virtual void Assign(void*, void const*) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Clone

```
public:virtual IATreeImpl* Clone() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

CreateNode

```
public:virtual void* CreateNode(void const*) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

CreateNodeWithStreamer

```
public:virtual void*
  CreateNodeWithStreamer( IACollectionStreamer*,
                          IDataStream& ) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

DeleteNode

```
public:virtual void DeleteNode(void*) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator INumber

```
public:virtual operator INumber() const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ReadTreeStructure

```
public:virtual void ReadTreeStructure(IDataStream&) = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

WriteTreeStructure

```
public:virtual void WriteTreeStructure(IDataStream&) const = 0
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

© Copyright IBM Corp. 2001

IAMultiwayTreeOps

```
protected:IAMultiwayTreeOps(bool)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IAMultiwayTreeOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IATreeOps

Class header file: iiatree.h

IATreeOps - Hierarchy List

IATreeOps

IATreeOps - Member Functions and Data by Group

Miscellaneous Members

~IATreeOps

```
public:virtual ~IATreeOps()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IATreeOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ICOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The ICOps Class is a combination of classes IStdMemOps, IStdAsOps, and IStdCmpOps; and therefore provides the memory management, and assignment and ordering relation element functions.

Class header file: istdops.h

ICOps - Hierarchy List

IStdMemOps
IStdOps
ICOps

ICOps - Member Functions and Data by Group

ICOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IECOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IECOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdEqOps, and IStdCmpOps; and therefore provides the memory management, and assignment, equality test, and ordering relation element functions.

Class header file: istdops.h

IECOps - Hierarchy List

IStdMemOps
  IStdOps
    IEOps
      IECOps

IECOps - Member Functions and Data by Group

IECOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEHOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdEqOps, and IStdHshOps; and therefore provides the memory management, and assignment, equality test, and hash element functions.

Class header file: istdops.h

IEHOps - Hierarchy List

IStdMemOps
  IStdOps
    IEOps

IEHOps

IEHOps - Member Functions and Data by Group

IEHOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEKCKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEKCKeyOps Class uses class IStdCmpOps; and therefore provides the ordering relation key-type functions.

Class header file: istdops.h

IEKCKeyOps - Hierarchy List

IStdCmpOps

IEKCKeyOps

IEKCKeyOps - Member Functions and Data by Group

IEKCKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IStdCmpOps			
Definition	**Page Number**	**Definition**	**Page Number**
`compare`	2798		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEKCOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEKCOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdEqOps, IStdKeyOps and IStdCmpOps; and therefore provides the memory management; assignment and equality test element functions; and ordering relation key-type functions.

Class header file: istdops.h

IEKCOps - Hierarchy List

IStdMemOps
  IStdOps
    IEOps
      IEKCOps

IEKCOps - Member Functions and Data by Group

Miscellaneous Members

keyOps

```
public:IEKCKeyOps < Key > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEKCOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEKEHKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEKEHKeyOps Class is a combination of classes IStdEqOps and IStdHshOps; and therefore provides the equality test and hash key-type functions.

Class header file: istdops.h

IEKEHKeyOps - Hierarchy List

IStdEqOps

IEKEHKeyOps

IEKEHKeyOps - Member Functions and Data by Group

IEKEHKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IStdEqOps			
Definition	**Page Number**	**Definition**	**Page Number**
`equal`	2799		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEKEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEKEHOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdEqOps, IStdKeyOps and IStdHshOps; and therefore provides the memory management; assignment and equality test element functions; and equality test and hash key-type functions.

Class header file: istdops.h

IEKEHOps - Hierarchy List

IStdMemOps
  IStdOps
    IEOps
      IEKEHOps

IEKEHOps - Member Functions and Data by Group

Miscellaneous Members

keyOps

```
public:IEKEHKeyOps < Key > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEKEHOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IEOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IEOps Class is a combination of classes IStdMemOps, IStdAsOps, and IStdEqOps; and therefore provides the memory management, and assignment and equality test element functions.

Class header file: istdops.h

IEOps - Hierarchy List

IStdMemOps
  IStdOps
    IEOps
      IECOps
      IEKEHOps
      IEKCOps
      IEHOps

IEOps - Member Functions and Data by Group

IEOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IKCKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IKCKeyOps Class uses class IStdCmpOps; and therefore provides the ordering relation key-type functions.

Class header file: istdops.h

IKCKeyOps - Hierarchy List

IStdCmpOps

IKCKeyOps

IKCKeyOps - Member Functions and Data by Group

IKCKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IStdCmpOps			
Definition	**Page Number**	**Definition**	**Page Number**
`compare`	2798		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IKCOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality

test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IKCOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdKeyOps and IStdCmpOps; and therefore provides the memory management; assignment element functions; and ordering relation key-type functions.

Class header file: istdops.h

IKCOps - Hierarchy List

IStdMemOps
IStdOps
IKCOps

IKCOps - Member Functions and Data by Group

Miscellaneous Members

keyOps

```
public:IKCKeyOps < Key > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKCOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IKEHKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IKEHKeyOps Class is a combination of classes IStdEqOps and IStdHshOps; and therefore provides the equality test and hash key-type functions.

Class header file: istdops.h

IKEHKeyOps - Hierarchy List

IStdEqOps
IKEHKeyOps

IKEHKeyOps - Member Functions and Data by Group

IKEHKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IStdEqOps			
Definition	**Page Number**	**Definition**	**Page Number**
`equal`	2799		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IKEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IKEHOps Class is a combination of classes IStdMemOps, IStdAsOps, IStdEqOps, IStdKeyOps and IStdHshOps; and therefore provides the memory management; assignment element functions; and equality test and hash key-type functions.

Class header file: istdops.h

IKEHOps - Hierarchy List

IStdMemOps

IStdOps

IKEHOps

IKEHOps - Member Functions and Data by Group

Miscellaneous Members

keyOps

```
public:IKEHKeyOps < Key > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IKEHOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IMultiwayTreeOps

Class header file: iimwt.h

IMultiwayTreeOps - Hierarchy List

IAMultiwayTreeOps

IMultiwayTreeOps

IMultiwayTreeOps - Member Functions and Data by Group

Miscellaneous Members

~IMultiwayTreeOps

```
public:~IMultiwayTreeOps()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assign

```
public:void Assign(void*, void const*) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Clone

```
public:IATreeImpl* Clone() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

CreateNode

```
public:void* CreateNode(void const*) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

CreateNodeWithStreamer

```
public:void*
  CreateNodeWithStreamer( IACollectionStreamer*,
                          IDataStream& ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

DeleteNode

```
public:void DeleteNode(void*) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

get_os_typespec

```
public:static os_typespec* get_os_typespec()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMultiwayTreeOps

Overload 1

```
public:IMultiwayTreeOps( IMultiwayTreeOps < numOfChildren,
                         Element,
                         ElementOps,
                         Implementation > const& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IMultiwayTreeOps()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IMultiwayTreeOps(void*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator Implementation *

```
public:operator Implementation *()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator INumber

```
public:operator INumber() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

os_mark_schema_type

```
public:static void os_mark_schema_type()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ReadTreeStructure

```
public:void ReadTreeStructure(IDataStream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

WriteTreeStructure

```
public:void WriteTreeStructure(IDataStream&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IMultiwayTreeOps - Inherited Member Functions and Data

Inherited Public Functions

IAMultiwayTreeOps			
Definition	**Page Number**	**Definition**	**Page Number**
~IAMultiwayTreeOps	2783	Assign	2783
Clone	2783	CreateNode	2783
CreateNodeWithStreamer	2783	DeleteNode	2783
operator INumber	2783	ReadTreeStructure	2783
WriteTreeStructure	2783		

Inherited Public Data

None

Inherited Protected Functions

IAMultiwayTreeOps			
Definition	**Page Number**	**Definition**	**Page Number**
IAMultiwayTreeOps	2784		

Inherited Protected Data

None

IStdAsOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdAsOps Class provides the assignment element function.

Class header file: istdops.h

IStdAsOps - Hierarchy List

IStdAsOps

IStdAsOps - Member Functions and Data by Group

Assignment Operation

Use this function to assign one collection element to another.

assign

```
public:void assign(Element&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the element passed in the second argument to the element specified in the first argument.

IStdAsOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdCmpOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdCmpOps Class provides the ordering relation element function.

Class header file: istdops.h

IStdCmpOps - Hierarchy List

IStdCmpOps

IEKCKeyOps

IKCKeyOps

IStdCmpOps - Member Functions and Data by Group

Compare Operation

Use this function to compare one collection element to another.

compare

```
public:long compare(Element const&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares two collection elements.

Returns

- -1. If the first element is less than the second element.
- 1. If the second element is less than the first element.
- 0. Otherwise

IStdCmpOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdEqOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdEqOps Class provides the equality test element function.

Class header file: istdops.h

IStdEqOps - Hierarchy List

IStdEqOps

IEKEHKeyOps

IKEHKeyOps

IStdEqOps - Member Functions and Data by Group

Equality Test Operation

Use this function to determine if two collection elements are equal.

equal

```
public:bool equal(Element const&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the first element equals the second element.

IStdEqOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdHshOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdHshOps Class provides the hash element function.

Class header file: istdops.h

IStdHshOps - Hierarchy List

IStdHshOps

IStdHshOps - Member Functions and Data by Group

Hash Operation

Use this function to obtain the hash value of a collection element.

hash

```
public:unsigned long hash(Element const&, unsigned long) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the hash value of the element.

IStdHshOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdKeyOps Class provides the key access element function.

Class header file: istdops.h

IStdKeyOps - Hierarchy List

IStdKeyOps

IStdKeyOps - Member Functions and Data by Group

Key Operation

Use this function to obtain the key for the element.

key

```
public:Key const& key(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the key of the specified element.

IStdKeyOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdMemOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdMemOps Class provides the memory management functions.

Class header file: istdops.h

IStdMemOps - Hierarchy List

IStdMemOps

IStdOps

IStdMemOps - Member Functions and Data by Group

Allocate Operation

Use this function to allocate memory for a collection.

allocate

```
public:void*
  allocate( size_t,
            char const*,
            unsigned long,
            os_typespec*,
            os_int32 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates space for dynamically created internal objects of a Collection Class (such as nodes in a tree or a linked list).

Deallocate Operation

Use this function to deallocate memory for a collection.

deallocate

```
public:void deallocate(void*, char const*, unsigned long) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes the space created by the allocate member function.

IStdMemOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IStdOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

The IStdOps Class is a combination the IStdMemOps Class and the IStdAsOps Class and therefore provides the memory management and assignment element functions.

Class header file: istdops.h

IStdOps - Hierarchy List

IStdMemOps

IStdOps

IKEHOps

IKCOps

IEOps

ICOps

IStdOps - Member Functions and Data by Group

IStdOps - Inherited Member Functions and Data

Inherited Public Functions

IStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`allocate`	2802	`deallocate`	2802

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWCOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWCOps Class is a combination of classes IWStdMemOps, IWStdAsOps, and IWStdCmpOps; and therefore provides the memory management, and assignment and ordering relation element functions.

Class header file: istdops.h

IWCOps - Hierarchy List

IWStdMemOps
  IWStdOps
    IWCOps

IWCOps - Member Functions and Data by Group

Miscellaneous Members

IWCOps

```
public:IWCOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWCOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
IWStdOps	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEHOps Class is a combination of classes IWStdMemOps, IWStdAsOps, IWStdEqOps, and IWStdHshOps; and therefore provides the memory management, and assignment, equality test, and hash element functions.

Class header file: istdops.h

IWEHOps - Hierarchy List

IWStdMemOps
  IWStdOps
    IWEOps
      IWEHOps

IWEHOps - Member Functions and Data by Group

Miscellaneous Members

IWEHOps

```
public:IWEHOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWEHOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWEOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWEOps`	2812		

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEKCKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEKCKeyOps Class uses class IWStdCmpKeyOps; and therefore provides the ordering relation key-type functions.

Class header file: istdops.h

IWEKCKeyOps - Hierarchy List

IWStdCmpKeyOps

IWEKCKeyOps

IWEKCKeyOps - Member Functions and Data by Group

IWEKCKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdCmpKeyOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Compare`	2819		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEKCOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of

an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEKCOps Class is a combination of classes IWStdMemOps, IWStdAsOps, IWStdEqOps, IWStdKeyOps and IWStdCmpOps; and therefore provides the memory management; assignment and equality test element functions; and ordering relation key-type functions.

Class header file: istdops.h

IWEKCOps - Hierarchy List

IWStdMemOps
  IWStdOps
    IWEOps
      IWEKCOps

IWEKCOps - Member Functions and Data by Group

Miscellaneous Members

IWEKCOps

```
public:IWEKCOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

keyOps

```
public:IWEKCKeyOps < _Key , ElementOps > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWEKCOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWEOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWEOps`	2812		

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEKEHKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEKEHKeyOps Class is a combination of classes IWStdEqKeyOps and IWStdHshKeyOps; and therefore provides the equality test and hash key-type functions.

Class header file: istdops.h

IWEKEHKeyOps - Hierarchy List

IWStdEqKeyOps

IWEKEHKeyOps

IWEKEHKeyOps - Member Functions and Data by Group

IWEKEHKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdEqKeyOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Equal`	2821		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEKEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEKEHOps Class is a combination of classes IWStdMemOps, IWStdAsOps, IWStdEqOps, IWStdKeyOps and IWStdHshOps; and therefore provides the memory management; assignment and equality test element functions; and equality test and hash key-type functions.

Class header file: istdops.h

IWEKEHOps - Hierarchy List

IWStdMemOps
  IWStdOps
    IWEOps
      IWEKEHOps

IWEKEHOps - Member Functions and Data by Group

Miscellaneous Members

IWEKEHOps

```
public:IWEKEHOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

keyOps

```
public:IWEKEHKeyOps < _Key , ElementOps > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWEKEHOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWEOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWEOps`	2812		

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWEOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWEOps Class is a combination of classes IWStdMemOps, IWStdAsOps, and IWStdEqOps; and therefore provides the memory management, and assignment and equality test element functions.

Class header file: istdops.h

IWEOps - Hierarchy List

IWStdMemOps
  IWStdOps
    IWEOps
      IWEHOps
      IWEKEHOps
      IWEKCOps

IWEOps - Member Functions and Data by Group

Miscellaneous Members

IWEOps

```
public:IWEOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWEOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWKCKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality

test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWKCKeyOps Class uses class IWStdCmpKeyOps; and therefore provides the ordering relation key-type functions.

Class header file: istdops.h

IWKCKeyOps - Hierarchy List

IWStdCmpKeyOps
IWKCKeyOps

IWKCKeyOps - Member Functions and Data by Group

IWKCKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdCmpKeyOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Compare`	2819		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWKCOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWKCOps Class is a combination of classes IWStdMemOps, IWStdAsOps, IWStdKeyOps and IWStdCmpOps; and therefore provides the memory management; assignment element functions; and ordering relation key-type functions.

Class header file: istdops.h

IWKCOps - Hierarchy List

IWStdMemOps

IWStdOps

IWKCOps

IWKCOps - Member Functions and Data by Group

Miscellaneous Members

IWKCOps

```
public:IWKCOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

keyOps

```
public:IWKCKeyOps < _Key , ElementOps > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWKCOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWKEHKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWKEHKeyOps Class is a combination of classes IWStdEqKeyOps and IWStdHshKeyOps; and therefore provides the equality test and hash key-type functions.

Class header file: istdops.h

IWKEHKeyOps - Hierarchy List

IWStdEqKeyOps

IWKEHKeyOps

IWKEHKeyOps - Member Functions and Data by Group

IWKEHKeyOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdEqKeyOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Equal`	2821		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWKEHOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWKEHOps Class is a combination of classes IWStdMemOps, IWStdAsOps, IWStdEqOps, IWStdKeyOps and IWStdHshOps; and therefore provides the memory management; assignment element functions; and equality test and hash key-type functions.

Class header file: istdops.h

IWKEHOps - Hierarchy List

IWStdMemOps

IWStdOps

IWKEHOps

IWKEHOps - Member Functions and Data by Group

Miscellaneous Members

IWKEHOps

```
public:IWKEHOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

keyOps

```
public:IWKEHKeyOps < _Key , ElementOps > keyOps
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWKEHOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

IWStdOps			
Definition	**Page Number**	**Definition**	**Page Number**
`IWStdOps`	2827		

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdAsOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdAsOps Class provides the assignment element function.

Class header file: istdops.h

IWStdAsOps - Hierarchy List

IWStdAsOps

IWStdAsOps - Member Functions and Data by Group

Assignment Operation

Use this function to assign one collection element to another.

Assign

```
public:void Assign(Element&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Assigns the element passed in the second argument to the element specified in the first argument by calling the assign member function of the supplied operations class.

IWStdAsOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdCmpKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdCmpKeyOps Class provides the ordering relation key-type function.

Class header file: istdops.h

IWStdCmpKeyOps - Hierarchy List

IWStdCmpKeyOps

IWEKCKeyOps

IWKCKeyOps

IWStdCmpKeyOps - Member Functions and Data by Group

Compare Operation

Use this function to compare one collection key to another.

Compare

```
public:long Compare(Key const&, Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares two collection keys by calling the compare member function of the supplied operations class.

Returns

- -1. If the first key is less than the second key.
- 1. If the second key is less than the first key.
- 0. Otherwise

IWStdCmpKeyOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdCmpOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations

Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdCmpOps Class provides the ordering relation element function.

Class header file: istdops.h

IWStdCmpOps - Hierarchy List

IWStdCmpOps

IWStdCmpOps - Member Functions and Data by Group

Compare Operation

Use this function to compare one collection element to another.

Compare

```
public:long Compare(Element const&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares two collection elements by calling the compare member function of the supplied operations class.

Returns

- -1. If the first element is less than the second element.
- 1. If the second element is less than the first element.
- 0. Otherwise

IWStdCmpOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdEqKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdEqKeyOps Class provides the equality test key-type function.

Class header file: istdops.h

IWStdEqKeyOps - Hierarchy List

IWStdEqKeyOps

IWEKEHKeyOps

IWKEHKeyOps

IWStdEqKeyOps - Member Functions and Data by Group

Equality Test Operation

Use this function to determine if two collection key elements are equal.

Equal

```
public:bool Equal(Key const&, Key const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the first key equals the second key. The keys are compared by calling the equal member function of the supplied operations class.

IWStdEqKeyOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdEqOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdEqOps Class provides the equality test element function.

Class header file: istdops.h

IWStdEqOps - Hierarchy List

IWStdEqOps

IWStdEqOps - Member Functions and Data by Group

Equality Test Operation

Use this function to determine if two collection elements are equal.

Equal

```
public:bool Equal(Element const&, Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the first element equals the second element. The elements are compared by calling the equal member function of the supplied operations class.

IWStdEqOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdHshKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdHshKeyOps Class provides the hash key-type function.

Class header file: istdops.h

IWStdHshKeyOps - Hierarchy List

IWStdHshKeyOps

IWStdHshKeyOps - Member Functions and Data by Group

Hash Operation

Use this function to obtain the hash value of a collection key.

Hash

```
public:unsigned long Hash(Key const&, unsigned long) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the hash value of the key by calling the hash member function of the supplied operations class.

IWStdHshKeyOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdHshOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdHshOps Class provides the hash element function.

Class header file: istdops.h

IWStdHshOps - Hierarchy List

IWStdHshOps

IWStdHshOps - Member Functions and Data by Group

Hash Operation

Use this function to obtain the hash value of a collection element.

Hash

```
public:unsigned long Hash(Element const&, unsigned long) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the hash value of the element by calling the hash member function of the supplied operations class.

IWStdHshOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdKeyOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdKeyOps Class provides the key access element function.

Class header file: istdops.h

IWStdKeyOps - Hierarchy List

IWStdKeyOps

IWStdKeyOps - Member Functions and Data by Group

Key Operation

Use this function to obtain the key for the element.

Key

```
public:_Key const& Key(Element const&) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the key of the specified element by calling the key member function of the supplied operations class

IWStdKeyOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdMemOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdMemOps Class provides the memory management functions.

Class header file: istdops.h

IWStdMemOps - Hierarchy List

IWStdMemOps

IWStdOps

IWStdMemOps - Member Functions and Data by Group

Allocate Operation

Use this function to allocate memory for a collection.

Allocate

```
public:void*
  Allocate( size_t,
            char const*,
            unsigned long,
            os_typespec*,
            os_int32 ) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates space for dynamically created internal objects of a Collection Class (such as nodes in a tree or a linked list) by calling the allocate member function of the supplied operations class.

Deallocate Operation

Use this function to deallocate memory for a collection.

Deallocate

```
public:void Deallocate(void*, char const*, unsigned long) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Deletes the space created by the allocate member function by calling the deallocate member function of the supplied operations class.

IWStdMemOps - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IWStdOps

An Element Operation Class used with IOC Collection Classes. An Element Operation Class provides one or more element functions, key-type functions, or memory management functions. The element functions that may be required by a Collection Class are: default and copy constructor, destructor, assignment, equality test, ordering relation, key access, and hash function. The key-type functions that may be required by a Collection Class are: equality test, ordering relation, and hash function.

Element Operation Classes can be inherited from to provide your own complete operations class, and are also used by the IOC Collection Classes themselves to provide default element operations for non-generic classes (generic classes begin with IG...) .

There are two types of Element Operations Classes those that begin IW... and those that begin I... . The I... member functions actually implement the Element Operations; whereas, the IW... member functions invoke the member functions of an I... type of class. Most IOC Collection Classes use the IW... Element Operations Classes. The non-generic IOC Collection Classes use the IW... classes in combination with the I... classes; whereas the generic IOC Collection Classes use the IW... with a user supplied version of the I... classes.

The IWStdOps Class is a combination the IWStdMemOps Class and the IWStdAsOps Class and therefore provides the memory management and assignment element functions.

Class header file: istdops.h

IWStdOps - Hierarchy List

IWStdMemOps

IWStdOps

IWEOps

IWCOps

IWKEHOps

IWKCOps

IWStdOps - Member Functions and Data by Group

Miscellaneous Members

IWStdOps

```
public:IWStdOps(void* = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IWStdOps - Inherited Member Functions and Data

Inherited Public Functions

IWStdMemOps			
Definition	**Page Number**	**Definition**	**Page Number**
`Allocate`	2826	`Deallocate`	2826

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 9. Math Classes

c_exception

Use the c_exception class to handle errors that are created by functions and operations in the complex class.

Note: The c_exception class is not related to the C++ exception handling mechanism that uses the **try**, **catch**, and **throw** statements.

Class header file: complex.h

c_exception - Hierarchy List

c_exception

c_exception - Member Functions and Data by Group

Constructors & Destructor

You can construct objects of this class.

There is no explicit c_exception destructor.

c_exception

```
public:c_exception( char* n,
             const complex& a1,
             const complex& a2 = complex_zero )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a c_exception object.

n The name of the function where the error occurred.

a1 The first complex argument with which the function that caused the error was called.

a2 The second complex argument with which the function that caused the error was called.

c_exception - Associated Globals

complex_error

```
int complex_error(c_exception&)
```

complex_error() is invoked by member functions of the Complex Mathematics Library when errors are detected. The argument refers to the c_exception object that contains information about the error. You can define your own procedures for handling errors by defining a function called complex_error() with return type int and a single parameter of type c_exception&.

Note: You can only override complex_error() if you are using the static version of the I/O Stream Library.

If you define your own complex_error() function and this function returns a nonzero value, no error message will be generated and the external

© Copyright IBM Corp. 2001

variable, errno, will not be set. If this function returns zero, errno is given the value of one of the following constants:

- ERANGE - if the result is too large or too small
- EDOM - if there is a domain error within a mathematical function

These constants are defined in errno.h

If you define your own version of complex_error(), when you must ensure that the name of the header file that contains your version of complex_error() is included in your source file when you compile your program.

Default Error-Handling Procedures

If you do not define your own complex_error(), the default error-handling procedures will be invoked when an error occurs. The results for a given input complex value (a, b) depend on the kind of error and the sign of the cosine and sine of b. The following table shows the return value of the default error-handling procedure and the value given to errno for each function with input equal to the complex value (a, b).

The following symbols appear in this table:

1. NA - not applicable. The result of the error depends on the sign of the cosine and sine of b (the imaginary part of the argument) unless ″NA″ appears in the Cosine b or Sine b columns.
2. HUGE - the maximum double value. This value is defined in math.h.

Function	Error	Cosine b	Sine b	Return Value	errno Value
cosh	a too large	non-negative	non-negative	(+HUGE, +HUGE)	ERANGE
cosh	a too large	non-negative	non-negative	(+HUGE, -HUGE)	ERANGE
cosh	a too small	non-negative	non-negative	(+HUGE, -HUGE)	ERANGE
cosh	a too small	non-negative	non-negative	(+HUGE, +HUGE)	ERANGE
cosh	a too small	negative	non-negative	(-HUGE, -HUGE)	ERANGE
cosh	a too small	negative	non-negative	(-HUGE, +HUGE)	ERANGE
cosh	b too large	negative	non-negative	(-HUGE, +HUGE)	ERANGE
cosh	b too large	negative	non-negative	(-HUGE, -HUGE)	ERANGE
cosh	b too small	NA	NA	(0, 0)	ERANGE
exp	a too large	positive	positive	(+HUGE, +HUGE)	ERANGE
exp	a too large	positive	positive	(+HUGE, -HUGE)	ERANGE
exp	a too large	non-positive	positive	(-HUGE, +HUGE)	ERANGE

Function	Error	Cosine b	Sine b	Return Value	errno Value
exp	a too large	non-positive	positive	(-HUGE, -HUGE)	ERANGE
exp	a too small	NA	NA	(0, 0)	ERANGE
exp	b too large	NA	NA	(0, 0)	ERANGE
exp	b too small	NA	NA	0, 0)	ERANGE
log	a too large	positive	positive	(+HUGE, 0)	EDOM (See note)
sinh	a too large	non-negative	non-negative	(+HUGE, +HUGE)	ERANGE
sinh	a too large	non-negative	negative	(+HUGE, -HUGE)	ERANGE
sinh	a too large	negative	non-negative	(-HUGE, +HUGE)	ERANGE
sinh	a too large	negative	negative	(-HUGE, -HUGE)	ERANGE
sinh	a too small	non-negative	non-negative	(-HUGE, +HUGE)	ERANGE
sinh	a too small	non-negative	negative	(-HUGE, -HUGE)	ERANGE
sinh	a too small	negative	non-negative	(+HUGE, +HUGE)	ERANGE
sinh	a too small	negative	negative	(+HUGE, -HUGE)	ERANGE
sinh	b too large	NA	NA	(0, 0)	ERANGE
sinh	b too small	NA	NA	(0, 0)	ERANGE

Note: errno is set to EDOM when the error for log() is detected.

Errors Not Handled

There are some cases where member functions of the Complex Mathematics Library call functions in the math library. These calls can cause underflow and overflow conditions that are handled by the matherr() function that is declared in the math.h header file. For example, the overflow conditions that are caused by the following calls are handled by matherr():

```
exp(complex(DBL_MAX, DBL_MAX))
pow(complex(DBL_MAX, DBL_MAX), INT_MAX)
norm(complex(DBL_MAX, DBL_MAX))
```

DBL_MAX is the maximum valid double value. INT_MAX is the maximum int value. Both these constants are defined in float.h.

If you do not want the default error-handling defined by matherr(), you should define your own version of matherr().

This is supported on **WIN OS/2 AIX OS/400 z/OS**

cosh

```
complex cosh(complex)
```

Returns the hyperbolic cosine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

cosh

```
complex cosh(const complex&)
```

Returns the hyperbolic cosine of the complex argument.

This is supported on **AIX**

coshhex

```
complex coshhex(complex)
```

Returns the hyperbolic cosine of the complex argument. This function is called by the cosh function when the application uses hexadecimal floating point and double values.

This is supported on **z/OS**

coshieee

```
complex coshieee(complex)
```

Returns the hyperbolic cosine of the complex argument. This function is called by the cosh function when the application uses IEEE floating point and double values.

This is supported on **z/OS**

exp

```
complex exp(complex)
```

Returns the complex value equal to e to the power of x where x is the argument.

This is supported on **WIN OS/2 OS/400 z/OS**

exp

```
complex exp(const complex&)
```

Returns the complex value equal to e to the power of x where x is the argument.

This is supported on **AIX**

exphex

```
complex exphex(complex)
```

Returns the complex value equal to e to the power of x where x is the argument. This function is called by the exp function when the application uses hexadecimal floating point and double values.

This is supported on **z/OS**

expieee

```
complex expieee(complex)
```

Returns the complex value equal to e to the power of x where x is the argument. This function is called by the exp function when the application uses IEEE floating point and double values.

This is supported on **z/OS**

log

```
complex log(complex)
```

Returns the natural logarithm of the argument x.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

loghex

```
complex loghex(complex)
```

Returns the natural logarithm of the argument x. This function is called by the exp function when the application uses hexadecimal floating point and double values.

This is supported on **z/OS**

logieee

```
complex logieee(complex)
```

Returns the natural logarithm of the argument x. This function is called by the exp function when the application uses IEEE floating point and double values.

This is supported on **z/OS**

sinh

```
complex sinh(complex)
```

Returns the hyperbolic sine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

sinh

```
complex sinh(const complex&)
```

Returns the hyperbolic sine of the complex argument.

This is supported on **AIX**

sinhhex

```
complex sinhhex(complex)
```

Returns the hyperbolic sine of the complex argument. This function is called by the sinh function when the application uses hexadecimal floating point and double values.

This is supported on **z/OS**

sinhieee

```
complex sinhieee(complex)
```

Returns the hyperbolic sine of the complex argument. This function is called by the sinh function when the application uses IEEE floating point and double values.

This is supported on **z/OS**

c_exception - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

complex

This class provides you with facilities to manipulate complex numbers.

A complex number is made up of two parts: a real part and an imaginary part. A complex number can be represented by an ordered pair (a, b), where a is the value of the real part of the number and b is the value of the imaginary part.

Class header file: complex.h

complex - Hierarchy List

complex

complex - Member Functions and Data by Group

Constructors & Destructor

These constructors can be used to create complex objects.

There is no explicit complex destructor.

Arrays of Complex Numbers

You can use the complex constructor to initialize arrays of complex numbers. If the list of initial values is made up of complex values, each array element is initialized to the corresponding value in the list of initial vlaues. If the list of initial values is not made up of complex values, the real parts of the array elements are initialized to these initial values and the imaginary parts of the array elements are initialized to 0.

In the following example, the elements of array b are initialized to the values in the initial value list, but only the real parts of elements of array a are initialized to the values in the initial value list.

```
#include < complex.h >

int main()
{
complex a[3] = {1.0, 2.0, 3.0};
complex b[3] = {complex(1.0, 1.0), complex(2.0, 2.0), complex(3.0, 3.0)};

cout <<  "Here is the first element of a:  " << a[0] << endl;
cout <<  "Here is the first element of b:  " << b[0] << endl;
}
```

This example produces the following output:

```
Here is the first element of a: ( 1, 0)
Here is the first element of b: ( 1, 1)
```

complex

Constructs a complex number.

Overload 1

```
public:complex(double r, double i = 0.0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a complex number.

The first argument, r, is assigned to the real part of the complex number. If you specify a second argument, it is assigned to the imaginary part of the complex number. If the second parameter is not specified, the imaginary part is initialized to 0.

Overload 2

```
public:complex()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a complex number . The real and imaginary parts of the complex number are initialized to (0, 0).

Assignment Operators

The assignment operators do not produce a value that can be used in an expression. The following code, for example, produces a compile-time error:

```
complex x, y, z; // valid declaration

x = (y += z); // invalid assignment causes a compile-time error

y += z; // correct method involves splitting expression
x = y; // into separate statements.
```

operator *=

Assigns the value of x * y to x.

Overload 1

```
public:void operator *=(const complex&)
```

This is supported on **AIX**

Overload 2

```
public:inline void operator *=(complex)
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator +=

Assigns the value of x + y to x.

Overload 1

```
public:inline void operator +=(complex)
```

This is supported on **WIN OS/2 OS/400 z/OS**

Overload 2

```
public:inline void operator +=(const complex&)
```

This is supported on **AIX**

operator -=

Assigns the value of x - y to x.

Overload 1

```
public:inline void operator -=(complex)
```

This is supported on **WIN OS/2 OS/400 z/OS**

Overload 2

```
public:inline void operator -=(const complex&)
```

This is supported on **AIX**

operator /=

Assigns the value of x / y to x.

Overload 1

```
public:inline void operator /=(complex)
```

This is supported on **WIN OS/2 OS/400 z/OS**

Overload 2

```
public:void operator /=(const complex&)
```

This is supported on **AIX**

Internal Functions

These functions are internal to the complex class and should not be used by application programs.

hexdiveq

```
public:void hexdiveq(complex)
```

This is supported on **z/OS**

An internal function called by operator/= when the application uses hexadecimal floating point and double values.

hexmuteq

```
public:void hexmuteq(complex)
```

This is supported on **z/OS**

An internal function called by operator*= when the application uses hexadecimal floating point and double values.

ieeediveq

```
public:void ieeediveq(complex)
```

This is supported on **z/OS**

An internal function called by operator/= when the application uses IEEE floating point and double values.

ieeemuteq

```
public:void ieeemuteq(complex)
```

This is supported on **z/OS**

An internal function called by operator*= when the application uses IEEE floating point and double values.

complex - Associated Globals

abs

```
double abs(complex)
```

Returns the absolute value or magnitude of its argument. The absolute value of a complex value (a, b) is the positive square root of $a^2 + b^2$.

This is supported on **WIN OS/2 OS/400 z/OS**

abs

```
double abs(const complex&)
```

Returns the absolute value or magnitude of its argument. The absolute value of a complex value (a, b) is the positive square root of $a^2 + b^2$.

This is supported on **AIX**

arg

```
double arg(complex)
```

Returns the angle (in radians) of the polar representation of its argument. If the argument is equal to the complex number (a, b), the angle returned is the angle in radians on the complex plane between the real axis and the vector (a, b). The return value has a range of -pi to pi.

This is supported on **WIN OS/2 OS/400 z/OS**

arg

```
double arg(const complex&)
```

Returns the angle (in radians) of the polar representation of its argument. If the argument is equal to the complex number (a, b), the angle returned is the angle in radians on the complex plane between the real axis and the vector (a, b). The return value has a range of -pi to pi.

This is supported on **AIX**

conj

```
complex conj(complex)
```

Returns the complex value equal to (a, -b) if the input argument is equal to (a, b).

This is supported on **WIN OS/2 OS/400 z/OS**

conj

```
inline complex conj(const complex&)
```

Returns the complex value equal to (a, -b) if the input argument is equal to (a, b).

This is supported on **AIX**

cos

```
complex cos(complex)
```

Returns the cosine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

cos

```
complex cos(const complex&)
```

Returns the cosine of the complex argument.

This is supported on **AIX**

cosh

```
complex cosh(complex)
```

Returns the hyperbolic cosine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

cosh

```
complex cosh(const complex&)
```

Returns the hyperbolic cosine of the complex argument.

This is supported on **AIX**

exp

```
complex exp(complex)
```

Returns the complex value equal to e to the power of x where x is the argument.

This is supported on **WIN OS/2 OS/400 z/OS**

exp

```
complex exp(const complex&)
```

Returns the complex value equal to e to the power of x where x is the argument.

This is supported on **AIX**

imag

```
double imag(const complex&)
```

Extracts the imaginary part of the complex number provided as the argument.

This is supported on **WIN OS/2 OS/400 z/OS**

imag

```
inline double imag(const complex&)
```

Extracts the imaginary part of the complex number provided as the argument.

This is supported on **AIX**

log

```
complex log(complex)
```

Returns the natural logarithm of the argument x.

This is supported on **AIX**

log

```
complex log(complex)
```

Returns the natural logarithm of the argument x.

This is supported on **WIN OS/2 OS/400 z/OS**

norm

```
double norm(complex)
```

Returns the square of the magnitude of its argument. If the argument x is equal to the complex number (a, b), norm() returns the value $a^2 + b^2$.

norm() is faster than abs(), but it is more likely to cause overflow errors.

This is supported on **WIN OS/2 OS/400 z/OS**

norm

```
double norm(const complex&)
```

Returns the square of the magnitude of its argument. If the argument x is equal to the complex number (a, b), norm() returns the value $a^2 + b^2$.

norm() is faster than abs(), but it is more likely to cause overflow errors.

This is supported on **AIX**

operator !=

```
int operator !=(complex, complex)
```

The inequality operator "!=" returns a nonzero value if x does not equal y. This operator tests for inequality by testing that the two real components are not equal and that the two imaginary components are not equal.

Because both components are double values, the inequality operator returns false only when both the real and imaginary components of the two values are identical. If you want an inequality operator that can test for an absolute difference within a certain tolerance between the two pairs of corresponding components, you can use a function such as the is_not_equal function.

This is supported on **WIN OS/2 OS/400 z/OS**

operator !=

```
inline int operator !=(const complex&, const complex&)
```

The inequality operator "!=" returns a nonzero value if x does not equal y. This operator tests for inequality by testing that the two real components are not equal and that the two imaginary components are not equal.

Because both components are double values, the inequality operator returns false only when both the real and imaginary components of the two values are identical. If you want an inequality operator that can test for an absolute difference within a certain tolerance between the two pairs of corresponding components, you can use a function such as the is_not_equal function.

This is supported on **AIX**

operator *

```
complex operator *(complex, complex)
```

The multiplication operator returns the product of x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **WIN OS/2 OS/400 z/OS**

operator *

```
complex operator *(const complex&, double)
```

The multiplication operator returns the product of x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator *

```
complex operator *(const complex&, const complex&)
```

The multiplication operator returns the product of x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator +

```
complex operator +(complex, complex)
```

The addition operator returns the sum of x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **WIN OS/2 OS/400 z/OS**

operator +

```
inline complex operator +(const complex&, const complex&)
```

The addition operator returns the sum of x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator -

```
inline complex operator -(const complex&, const complex&)
```

The subtraction operator returns the difference between x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator -

```
complex operator -(complex, complex)
```

The subtraction operator returns the difference between x and y.

This operator has the same precedence as the corresponding real operator.

This is supported on **WIN OS/2 OS/400 z/OS**

operator -

```
inline complex operator -(const complex&)
```

The negation operator returns (-a, -b) when its argument is (a, b).

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator -

```
complex operator -(complex)
```

The negation operator returns (-a, -b) when its argument is (a, b).

This operator has the same precedence as the corresponding real operator.

This is supported on **WIN OS/2 OS/400 z/OS**

operator /

```
complex operator /(const complex&, double)
```

The division operator returns the quotient of x divided by y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator /

```
complex operator /(const complex&, const complex&)
```

The division operator returns the quotient of x divided by y.

This operator has the same precedence as the corresponding real operator.

This is supported on **AIX**

operator /

```
complex operator /(complex, complex)
```

The division operator returns the quotient of x divided by y.

This operator has the same precedence as the corresponding real operator.

This is supported on **WIN OS/2 OS/400 z/OS**

operator ==

```
int operator ==(complex, complex)
```

The equality operator ″==″ returns a nonzero value if x equals y. This operator tests for equality by testing that the two real components are equal and that the two imaginary components are equal.

Because both components are double values, the equality operator tests for an exact match between the two sets of values. If you want an equality operator that can test for an absolute difference within a certain tolerance between the two pairs of corresponding components, you can use a function such as the isequal function.

This is supported on **WIN OS/2 OS/400 z/OS**

operator ==

```
inline int operator ==(const complex&, const complex&)
```

The equality operator ″==″ returns a nonzero value if x equals y. This operator tests for equality by testing that the two real components are equal and that the two imaginary components are equal.

Because both components are double values, the equality operator tests for an exact match between the two sets of values. If you want an equality operator that can test for an absolute difference within a certain tolerance between the two pairs of corresponding components, you can use a function such as the isequal function.

This is supported on **AIX**

polar

```
complex polar(double, double = 0)
```

Returns the standard complex representation of the complex number that has a polar representation (a, b).

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pow

```
complex pow(complex, double)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **WIN OS/2 OS/400 z/OS**

pow

```
complex pow(double, complex)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **WIN OS/2 OS/400 z/OS**

pow

```
complex pow(complex, complex)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **WIN OS/2 OS/400 z/OS**

pow

```
complex pow(complex, int)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **WIN OS/2 OS/400 z/OS**

pow

```
complex pow(const complex&, int)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **AIX**

pow

```
complex pow(const complex&, double)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **AIX**

pow

```
complex pow(const complex&, const complex&)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **AIX**

pow

```
complex pow(double, const complex&)
```

Returns the complex value x^y, where x is the first argument and y is the second argument.

This is supported on **AIX**

real

```
double real(const complex&)
```

Extracts the real part of the complex number provided as the argument.

This is supported on **WIN OS/2 OS/400 z/OS**

real

```
inline double real(const complex&)
```

Extracts the real part of the complex number provided as the argument.

This is supported on **AIX**

sin

```
complex sin(const complex&)
```

Returns the sine of the complex argument.

This is supported on **AIX**

sin

```
complex sin(complex)
```

Returns the sine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

sinh

```
complex sinh(const complex&)
```

Returns the hyperbolic sine of the complex argument.

This is supported on **AIX**

sinh

```
complex sinh(complex)
```

Returns the hyperbolic sine of the complex argument.

This is supported on **WIN OS/2 OS/400 z/OS**

sqrt

```
complex sqrt(complex)
```

Returns the square root of its argument. If c and d are real values, then every complex number (a, b), where:

$a = c^2 - d^2$
$b = 2cd$

has two square roots:

(c, d)
(-c, -d)

sqrt() returns the square root that has a positive real part, that is, the square root that is contained in the first or fourth quadrants of the complex plane.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

complex - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IBinaryCodedDecimal

Objects of the IBinaryCodedDecimal class represent numerical quantities. Use them in business and commercial applications for accurate financial calculations.

This class provides functions to obtain information about, compare, and manipulate IBinaryCodedDecimal objects.

IBinaryCodedDecimal uses the utility class IDecimalUtil to perform its operations.

Class header file: idecimal.hpp

IBinaryCodedDecimal - Hierarchy List

IBinaryCodedDecimal

IBinaryCodedDecimal - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IBinaryCodedDecimal objects.

IBinaryCodedDecimal

The IBinaryCodedDecimal object can be constructed in many different ways. If the number of digits and precisions are provided, the constructed object has the provided digit and precision; otherwise default values are assumed according to the type of the value. If a value is provided during construction, the constructed object has the given value with the appropriate digit and precision; otherwise the constructed object will have a value zero. An exception is thrown if the number of digits in the integral part is larger than the supported value (for example, DEC_DIG) or the provided digit and/or precision are out of range. The valid values should satisfy the following:

```
DIGITS - the given number of digits
PRECISIONS - the given number of precisions

  0  <   DIGITS      <=  DEC_DIG
  0  <=  PRECISIONS  <=  DEC_DIG
 PRECISIONS  <=  DIGITS
```

DEC_DIG is defined in idecimal.hpp

Overload 1

```
public:IBinaryCodedDecimal(int anInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **anInt** with DFT_INT_DIG number of digits and zero number of precisions.

DFT_INT_DIG is defined in idecimal.hpp.

Overload 2

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    unsigned long auLong )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auLong** with **nDig** number of digits and **nPrec** number of precisions.

Overload 3

```
public:IBinaryCodedDecimal()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a default constructor. The constructed object has a value zero with attributes of DFT_DIG number of digits and DFT_PREC number of precisions.

Overload 4

```
public:IBinaryCodedDecimal(long double alDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **alDouble**.

Note: Precision loss could occur during conversion.

Overload 5

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    long aLong )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aLong** with **nDig** number of digits and **nPrec** number of precisions.

Overload 6

```
public:IBinaryCodedDecimal(float afloat)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **afloat**.

Note: Precision loss could occur during conversion.

Overload 7

```
public:IBinaryCodedDecimal(unsigned long auLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auLong** with DFT_LNG_DIG number of digits and zero number of precisions.

DFT_LNG_DIG is defined in idecimal.hpp.

Overload 8

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    long double alDouble )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **alDouble** with **nDig** number of digits and **nPrec** number of precisions.

Note: Precision loss could occur during conversion.

Overload 9

```
public:IBinaryCodedDecimal(unsigned int nDig, unsigned int nPrec)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value zero with **nDig** number of digits and **nPrec** number of precisions.

Overload 10

```
public:IBinaryCodedDecimal(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aDouble**.

Note: Precision loss could occur during conversion.

Overload 11

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    float afloat )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **afloat** with **nDig** number of digits and **nPrec** number of precisions.

Note: Precision loss could occur during conversion.

Overload 12

```
public:IBinaryCodedDecimal(const char* val)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object according to the value represented in the provided string. The number of digits and precisions are set accordingly. The leading and trailing zeros are used when setting up the digit and precision attributes, for example, ″00123.4560″ has 9 digits and 4 precisions.

Overload 13

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    int anInt )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **anInt** with **nDig** number of digits and **nPrec** number of precisions.

Overload 14

```
public:IBinaryCodedDecimal(unsigned long long auLongLong)
```

This is supported on **AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auLongLong** with DFT_LNG_DIG number of digits and zero number of precisions.

DFT_LNG_DIG is defined in idecimal.hpp.

Note: This constructor is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 15

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    unsigned long long auLongLong )
```

This is supported on **AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auLongLong** with **nDig** number of digits and **nPrec** number of precisions.

Note: This constructor is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 16

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    long long aLongLong )
```

This is supported on **AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aLongLong** with **nDig** number of digits and **nPrec** number of precisions.

Note: This constructor is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 17

```
public:IBinaryCodedDecimal(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aLongLong** with DFT_LNG_DIG number of digits and zero number of precisions.

DFT_LNG_DIG is defined in idecimal.hpp.

Note: This constructor is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 18

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    double aDouble )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aDouble** with **nDig** number of digits and **nPrec** number of precisions.

Note: Precision loss could occur during conversion.

Overload 19

```
public:IBinaryCodedDecimal(unsigned int auInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auInt** with DFT_INT_DIG number of digits and zero number of precisions.

DFT_INT_DIG is defined in idecimal.hpp.

Overload 20

```
public:IBinaryCodedDecimal( unsigned int nDig,
                    unsigned int nPrec,
                    unsigned int auInt )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **auInt** with **nDig** number of digits and **nPrec** number of precisions.

Overload 21

```
public:IBinaryCodedDecimal(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs the IBinaryCodedDecimal object to have a value **aLong** with DFT_LNG_DIG number of digits and zero number of precisions.

DFT_LNG_DIG is defined in idecimal.hpp.

Assignment Operators

These operators allow you to assign various forms of numbers to a IBinaryCodedDecimal object.

operator =

Assigns a new value to this object.

Overload 1

```
public:IBinaryCodedDecimal& operator =(unsigned auInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IBinaryCodedDecimal& operator =(int anInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IBinaryCodedDecimal& operator =(long double alDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IBinaryCodedDecimal&
  operator =( const IBinaryCodedDecimal& bcd )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:IBinaryCodedDecimal& operator =(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 6

```
public:IBinaryCodedDecimal& operator =(const IDecimalUtil& dut)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 7

```
public:IBinaryCodedDecimal&
  operator =( unsigned long long auLongLong )
```

This is supported on **AIX OS/400 z/OS**

Note: This operator is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 8

```
public:IBinaryCodedDecimal& operator =(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

Note: This operator is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

Overload 9

```
public:IBinaryCodedDecimal& operator =(unsigned long auLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 10

```
public:IBinaryCodedDecimal& operator =(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 11

```
public:IBinaryCodedDecimal& operator =(float aFloat)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Comparison Operators

Use these operators to compare IBinaryCodedDecimal objects for equality and/or relative magnitude.

operator !=

```
public:int operator !=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the represented values in both objects are not the same; otherwise returns 0.

operator <

```
public:int operator <(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the left operand represents a smaller value than the right operand; otherwise returns 0.

operator <=

```
public:int operator <=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the left operand represents a smaller or same value than the right operand; otherwise returns 0.

operator ==

```
public:int operator ==(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the represented values in both object are the same; otherwise returns 0.

operator >

```
public:int operator >(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the left operand represents a larger value than the right operand; otherwise returns 0.

operator >=

```
public:int operator >=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to compare two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

Returns 1 if the left operand represents a larger or same value than the right operand; otherwise returns 0.

Manipulations

Use these operators and member functions to perform common operations on IBinaryCodedDecimal objects.

operator !

```
public:int operator !() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns 1 if the IBinaryCodedDecimal object contains the value zero; otherwise returns 0.

operator *=

```
public:void operator *=(const IDecimalUtil& dut)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to to perform compound assignment between two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

The left operand is multiplied by the right operand, the result is stored back to the left operand.

operator +

```
public:IBinaryCodedDecimal operator +() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a unary plus operator. The copy of the original object is returned.

operator ++

This is a increment operator.

Overload 1

```
public:IBinaryCodedDecimal operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a prefix increment operator. The value being represented is incremented by 1 and the object with the new value is returned.

Overload 2

```
public:IBinaryCodedDecimal operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a postfix increment operator. The original object is returned. As a side effect, the value being represented is incremented by 1.

operator +=

```
public:void operator +=(const IDecimalUtil& dut)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to to perform compound assignment between two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

The right operand is added to the left operand, the result is stored back to the left operand.

operator -

```
public:IBinaryCodedDecimal operator -() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a unary minus operator. The negation of the object is returned.

operator --

This is a decrement operator.

Overload 1

```
public:IBinaryCodedDecimal operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a postfix decrement operator. The original object is returned. As a side effect, the value being represented is decremented by 1.

Overload 2

```
public:IBinaryCodedDecimal operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This is a prefix decrement operator. The value being represented is decremented by 1 and the object with the new value is returned.

operator -=

```
public:void operator -=(const IDecimalUtil& dut)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to to perform compound assignment between two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

The left operand is subtracted from the right operand, the result is stored back to the left operand.

operator /=

```
public:void operator /=(const IDecimalUtil& dut)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This operator is used to to perform compound assignment between two IBinaryCodedDecimal objects. The right IBinaryCodedDecimal object is converted to the utility class, IDecimalUtil first.

The left operand is divided by the right operand, the result is stored back to the left operand.

Queries

Use these member functions to query the object for status information.

cData

```
public:const char* cData() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer that points to the internal buffer which contains the binary coded decimal value in packed decimal format, for example two digits per byte, the last right nibble contains the sign.

This expression

```
(digitsOf() >> 1) + 1
```

identifies the number of bytes to store the binary coded decimal value.

Note: The buffer is not null terminated.

digitsOf

```
public:int digitsOf() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of digits specified when constructing the IBinaryCodedDecimal object. If it is not specified, the default is used, which does not necessarily correspond to the actual number of digits of the number.

isNegative

```
public:const bool isNegative() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the IBinaryCodeDecimal object represents a negative value; otherwise returns false.

isPositive

```
public:const bool isPositive() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the IBinaryCodeDecimal object represents a positive value; otherwise returns false.

Note: Zero is considered a positive value.

precisionOf

```
public:int precisionOf() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the number of precisions specified when constructing the IBinaryCodedDecimal object. If it is not specified, the default is used, which does not necessarily correspond to the actual number of precisions of the number.

Streaming

These operators and member functions support binary streaming of IBinaryCodedDecimal objects.

operator <<=

```
public:void operator <<=(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The value and attributes are changed by the input data stream, **fromWhere**.

operator >>=

```
public:void operator >>=(IDataStream& toWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The information of this object is streamed out to the **toWhere** data stream.

readFromStream

```
protected:void readFromStream(IDataStream& fromWhere)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Classes that inherit from IBinaryCodedDecimal can call readFromStream() to get information from the data stream **fromWhere** for the IBinaryCodedDecimal class.

writeToStream

```
protected:void writeToStream(IDataStream& toWhere) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Classes that inherit from IBinaryCodedDecimal can call writeToStream() to output the information for the IBinaryCodedDecimal class to the data stream **toWhere**.

Type Conversions

These member functions return the IBinaryCodedDecimal value as binary or text.

asDouble

```
public:double asDouble() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the binary coded decimal as a double value.

asLong

```
public:long asLong() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the binary coded decimal as a long type value. Any numbers after the decimal point are truncated. If the value in the binary coded decimal is too large or too small to be represented as a long type, the returned value is undefined.

asLongLong

```
public:long long asLongLong() const
```

This is supported on **AIX OS/400 z/OS**

Returns the binary coded decimal as a long long type value. Any numbers after the decimal point are truncated. If the value in the binary coded decimal is too large or too small to be represented as a long long type, the returned value is undefined.

Note: This function is available if IC_LONG_LONG is defined for the platform. This value is defined in icomdefs.h.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the binary coded decimal as an IString representation.

IBinaryCodedDecimal - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IDecimal

Use the IDecimal class to improve the performance of our applications relative to using the IBinaryCodeDecimal class. The IDecimal class is compatible with the decimal data type in C. This class permits you to represent up to 31 significant digits, including integral and fractional parts.

You can declare typedefs, arrays, and structures that have IDecimal objects. You can apply arithmetic, relational, assignment, equality, and unary operators on the IDecimal object. You can pass IDecimal objects in function calls.

Class header file: idecimal.hpp

IDecimal - Hierarchy List

IDecimalBase

IDecimal

IDecimal - Member Functions and Data by Group

Constructors & Destructor

Use these functions to construct IDecimal objects. The IDecimal object can be constructed in many different ways. The number of digits must be specified. Specifying the precision is optional; the default value for precision is 0. The constructed object will have either the provided number of digits and precision or the provided number of digits and 0 precision.

If a value for the object is provided during construction, the constructed object has the given value with the appropriate digit and precision; otherwise the constructed object will have a value of zero.

The range of digits and precision is:

```
DIGITS - the given number of digits (integral + any decimal digits excluding
                 sign and decimal points)
PRECISIONS - the given number of precisions (i.e. decimal digits)

  1  <=   DIGITS     <=  31
  0  <=  PRECISIONS  <=  DIGITS

Specifying a value for PRECISION is optional; the default is zero.
```

Decimal numbers must be in quotation marks.

The number of integral digits cannot be greater than

```
  DIGITS - PRECISION
```

otherwise an exception is thrown.

If the number of decimal digits is greater than PRECISION, extra digits to the right of the decimal parts will be dropped with no rounding.

If the provided digits and/or precision are out of range, the maximum values are used an no exception is thrown. For example:

```
  IDecimal<36,34>op1  ===>  IDecimal<31,31>op1
  IDecimal<5,29>op1   ===>  IDecimal<5,5>op1
  IDecimal<36,5>op1   ===>  IDecimal<31,5>op1
```

IDecimal

Overload 1

```
public:IDecimal(const char*)
```

This is supported on **z/OS**

Constructs the IDecimal object according to the value represented in the provided string. The number of digits and precision are set accordingly. The leading and trailing zeros are used when setting up the digit and precision attributes, for example, ″00123.4560″ has 9 digits including 4 digits of precision.

Overload 2

```
public:IDecimal(const IBinaryCodedDecimal)
```

This is supported on **z/OS**

Constructs the IDecimal object with nDig number of digits and nPrec digits of precisions by accepting an IBinaryCodedDecimal type value.

Overload 3

```
public:IDecimal(const int)
```

This is supported on **z/OS**

Constructs the IDecimal object with nDig number of digits and nPrec digits of precisions by accepting a value of type int.

Overload 4

```
public:IDecimal(const IDecimalProxy)
```

This is supported on **z/OS**

Constructs the IDecimal object with nDig number of digits and nPrec digits of precisions by accepting a IDecimalProxy type value.

Overload 5

```
public:IDecimal()
```

This is supported on **z/OS**

Constructs the IDecimal object to have a value of zero with nDig number of digits and nPrec digits of precisions.

Overload 6

```
public:IDecimal(const IDecimalResult)
```

This is supported on **z/OS**

Constructs the IDecimal object with nDig number of digits and nPrec digits of precisions by accepting a IDecimalResult type value.

Assignment Operators

Use these operators to assign different data types to the IDecimal object.

operator =

Overload 1

```
public:void operator =(const IDecimalResult&)
```

This is supported on **z/OS**

Overload 2

```
public:void operator =(const IDecimalProxy&)
```

This is supported on **z/OS**

Manipulation Operators and Functions

Use these operators to manipulate the IDecimal object.

operator *=

This operator is used to perform compound assignment. The left operand is multiplied by the right operand, the result is stored back to the left operand.

Overload 1

```
public:inline IDecimal < w , p >& operator *=(int)
```

This is supported on **z/OS**

Overload 2

```
public:IDecimal < w , p >&
  operator *=( const IDecimalProxy& )
```

This is supported on **z/OS**

operator ++

This is an increment operator.

Overload 1

```
public:inline IDecimal < w , p >& operator ++()
```

This is supported on **z/OS**

Overload 2

```
public:inline IDecimalResult operator ++(int)
```

This is supported on **z/OS**

operator +=

This operator is used to perform compound assignment. The right operand is added to the left operand, the result is stored back to the left operand.

Overload 1

```
public:inline IDecimal < w , p >& operator +=(const int)
```

This is supported on **z/OS**

Overload 2

```
public:IDecimal < w , p >&
  operator +=( const IDecimalProxy& )
```

This is supported on **z/OS**

operator --

This is an decrement operator.

Overload 1

```
public:inline IDecimalResult operator --(int)
```

This is supported on **z/OS**

Overload 2

```
public:inline IDecimal < w , p >& operator --()
```

This is supported on **z/OS**

operator -=

This operator is used to perform compound assignment. The right operand is subtracted from the left operand, the result is stored back to the left operand.

Overload 1

```
public:inline IDecimal < w , p >& operator -=(const int)
```

This is supported on **z/OS**

Overload 2

```
public:IDecimal < w , p >&
  operator -=( const IDecimalProxy& )
```

This is supported on **z/OS**

operator /=

This operator is used to perform compound assignment. The left operand is divided by the right operand, the result is stored back to the left operand.

Overload 1

```
public:inline IDecimal < w , p >& operator /=(int)
```

This is supported on **z/OS**

Overload 2

```
public:IDecimal < w , p >&
  operator /=( const IDecimalProxy& )
```

This is supported on **z/OS**

Query Functions

Use these functions to obtain information about the IDecimal object.

cData

```
public:const char* cData(void) const
```

This is supported on **z/OS**

Returns a pointer that points to the internal buffer which contains the decimal value in packed decimal format, for example two digits per byte, the last right nibble contains the sign.

The expression (digitsOf() >> 1) + 1 identifies the number of bytes to store the decimal value.

Note: The buffer is not null terminated.

digitsOf

```
public:int digitsOf(void) const
```

This is supported on **z/OS**

Returns the number of digits represented by the decimal value.

isNegative

```
public:int isNegative(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimal object represents a negative value; otherwise returns zero (0).

isPositive

```
public:int isPositive(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimal object represents a positive value; otherwise returns zero (0).

left

```
public:int left(void) const
```

This is supported on **z/OS**

Returns the value to the left of the decimal point, or width - precision.

precisionOf

```
public:int precisionOf(void) const
```

This is supported on **z/OS**

Returns the number of digits of precision.

right

```
public:int right(void) const
```

This is supported on **z/OS**

Returns the precision value of the IDecimal object.

Type Conversion Functions

Use these functions to convert the IDecimal object from one type to another.

asBCD

```
public:IBinaryCodedDecimal asBCD(void) const
```

This is supported on **z/OS**

Returns the decimal object as an IBinaryCodedDecimal representation.

asString

```
public:IString asString(void) const
```

This is supported on **z/OS**

Returns the IDecimal object as an IString representation. The sign (+ or -) of the decimal is also included, and the number is zero filled to the specified DIGITS and PRECISION. If a sign was not specified when constructing, + sign is the default.

operator IDecimalProxy

```
public:operator IDecimalProxy() const
```

This is supported on **z/OS**

IDecimal - Enumerations

Variation 1

```
enum  { width=(w>IDecimalBase::maxwidth)?IDecimalBase::maxwidth:w }
```

This is supported on **z/OS**

Variation 2

```
enum  { precision=(p<0)?0:(p>width)?width:p }
```

This is supported on **z/OS**

IDecimal - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IDecimalBase			
Definition	**Page Number**	**Definition**	**Page Number**
AP	2861	APE	2862
APEC	2862	COMPEQ	2862
COMPGE	2862	COMPGT	2862
COMPLE	2862	COMPLT	2862
CVB	2862	CVBS	2862
CVD	2862	CVDE	2862
DP	2862	DPE	2862
DPEC	2862	EDIT	2863
EDITMK	2863	MP	2863
MPE	2863	MPEC	2863
PACK	2863	proxyPB	2864
resultPB	2864	SP	2863
SPE	2863	SPEC	2863
SRP	2863	SRPE	2863
SRPR	2863	SRPRE	2863
UNPACK	2864	ZAP	2864
ZAPE	2864	ZAPEC	2864

Inherited Protected Data

None

IDecimalBase

This class is designed for internal use only and is not supported.

Class header file: idecimal.hpp

IDecimalBase - Hierarchy List

IDecimalBase

IDecimal

IDecimalResult

IDecimalProxy

IDecimalBase - Member Functions and Data by Group

Assembler Functions

AP

```
protected:static void AP(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

APE

```
protected:static void APE(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

APEC

```
protected:static void APEC(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

COMPEQ

```
protected:static int COMPEQ(unsigned int lens, const char* s)
```

This is supported on **z/OS**

COMPGE

```
protected:static int COMPGE(unsigned int lens, const char* s)
```

This is supported on **z/OS**

COMPGT

```
protected:static int COMPGT(unsigned int lens, const char* s)
```

This is supported on **z/OS**

COMPLE

```
protected:static int COMPLE(unsigned int lens, const char* s)
```

This is supported on **z/OS**

COMPLT

```
protected:static int COMPLT(unsigned int lens, const char* s)
```

This is supported on **z/OS**

CVB

```
protected:static int CVB(unsigned int len, char* t)
```

This is supported on **z/OS**

CVBS

```
protected:static int CVBS(unsigned int len, char* t, int amt)
```

This is supported on **z/OS**

CVD

```
protected:static void CVD(unsigned int len, char* t, int val)
```

This is supported on **z/OS**

CVDE

```
protected:static void CVDE(unsigned int len, char* t, int val)
```

This is supported on **z/OS**

DP

```
protected:static void DP(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

DPE

```
protected:static void DPE(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

DPEC

```
protected:static void DPEC(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

EDIT

```
protected:static void EDIT(unsigned int len, char* p, const char* s)
```

This is supported on **z/OS**

EDITMK

```
protected:static char*
  EDITMK( unsigned int len,
          char* p,
          const char* s )
```

This is supported on **z/OS**

MP

```
protected:static void MP(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

MPE

```
protected:static void MPE(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

MPEC

```
protected:static void MPEC(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

PACK

```
protected:static void PACK(unsigned int lens, char* p, const char* s)
```

This is supported on **z/OS**

SP

```
protected:static void SP(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

SPE

```
protected:static void SPE(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

SPEC

```
protected:static void SPEC(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

SRP

```
protected:static void SRP(unsigned int len, char* t, int amt)
```

This is supported on **z/OS**

SRPE

```
protected:static void SRPE(unsigned int len, char* t, int amt)
```

This is supported on **z/OS**

SRPR

```
protected:static void SRPR(unsigned int len, char* t, int amt)
```

This is supported on **z/OS**

SRPRE

```
protected:static void SRPRE(unsigned int len, char* t, int amt)
```

This is supported on **z/OS**

UNPACK

```
protected:static void
  UNPACK( unsigned int lens,
          char* p,
          const char* s )
```

This is supported on **z/OS**

ZAP

```
protected:static void ZAP(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

ZAPE

```
protected:static void ZAPE(unsigned int lens, char* t, const char* s)
```

This is supported on **z/OS**

ZAPEC

```
protected:static void
  ZAPEC( unsigned int lens,
         char* t,
         const char* s )
```

This is supported on **z/OS**

Query Functions

proxyPB

```
protected:inline static const char* proxyPB(const IDecimalProxy&)
```

This is supported on **z/OS**

resultPB

Returns the decimal value as a character string.

Overload 1

```
protected:inline static const char* resultPB(const IDecimalResult&)
```

This is supported on **z/OS**

Overload 2

```
protected:inline static char* resultPB(IDecimalResult&)
```

This is supported on **z/OS**

IDecimalBase - Enumerations

Variation 1

```
enum  { plusSign=0x0c,
        minusSign=0x0d }
```

This is supported on **z/OS**

Variation 2

```
enum  { maxbytes=16 }
```

This is supported on **z/OS**

Variation 3

```
enum  { maxwidth=31 }
```

This is supported on **z/OS**

IDecimalBase - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IDecimalDataError

The IDecimalDataError class represents an exception. Whenever a IBinaryCodedDecimal object or IDecimalUtil object encounters an underflow or overflow condition, it creates and throws an object of the IDecimalDataError class. These errors can occur while constructing a IBinaryCodedDecimal or IDecimalUtil object from a number or while performing arithmatic operations on these objects.

Class header file: iexcbase.hpp

IDecimalDataError - Hierarchy List

IException

IDecimalDataError

IDecimalDataError - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and destruct objects of this class. You cannot assign one IDecimalDataError object from another.

~IDecimalDataError

```
public:virtual ~IDecimalDataError()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IDecimalDataError

You can create objects of this class by doing the following:

- Using the constructor where you specify the error text, error ID, and severity.
- Using the macros discussed in IException. The Open Class Library provides these macros to make creating exceptions easier for you.

You can also copy an instance of this exception class.

Overload 1

```
public:IDecimalDataError( const char* errorText,
                          ioc::unumeric_t errorId = 0,
                          Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 2

```
public:IDecimalDataError( const ioc::unichar_t* errorText,
                  ioc::unumeric_t errorId = 0,
                  Severity severity = IException::unrecoverable )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructor.

errorText
The text describing this particular error.

errorId
The identifier you want to associate with this particular error.

severity
Use the enumeration IException::Severity to specify the severity of the error. The default is unrecoverable.

Overload 3

```
public:IDecimalDataError(const IDecimalDataError& exception)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The copy constructor is provided so that the compiler can make copies of the object during the throwing of an exception.

Exception Type

These member functions provide support for determining the name (type) of the exception. They are used for logging out an exception object's error information.

name

```
public:virtual const char* name() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

nameW

```
public:virtual const ioc::unichar_t* nameW() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the name of the class as a Unicode string. This member function is intended to be used by general exception handling code that does not have prior knowledge of the actual derived class type.

IDecimalDataError - Inherited Member Functions and Data

Inherited Public Functions

IException			
Definition	**Page Number**	**Definition**	**Page Number**
~IException	765	addLocation	767
appendText	769	appendTextW	769
assertParameter	771	assertParameterW	771
errorCodeGroup	766	errorId	767
IException	765	isRecoverable	769
locationAtIndex	768	locationCount	768
logExceptionData	768	logExceptionDataW	768
setErrorCodeGroup	766	setErrorId	767
setSeverity	769	setText	769
setTextW	769	setTraceFunction	768
terminate	765	text	770
textCount	770	textW	770

Inherited Public Data

IException			
Definition	**Page Number**	**Definition**	**Page Number**
baseLibrary	766	CLibrary	766
operatingSystem	766	other	767
presentationSystem	767		

Inherited Protected Functions

None

Inherited Protected Data

None

IDecimalProxy

This class is designed for internal use only and is not supported.

Class header file: idecimal.hpp

IDecimalProxy - Hierarchy List

IDecimalBase

IDecimalProxy

IDecimalProxy - Member Functions and Data by Group

Constructors & Destructor

Use these functions to construct IDecimalProxy objects.

IDecimalProxy

```
public:inline IDecimalProxy(const char* d, unsigned int w, int p)
```

This is supported on **z/OS**

Constructs an object of this class.

Data Members

precision

```
public:const int precision
```

This is supported on **z/OS**

width

```
public:const unsigned int width
```

This is supported on **z/OS**

Query Functions

Use these functions to obtain information about the IDecimalProxy object.

cData

```
public:const char* cData(void) const
```

This is supported on **z/OS**

Returns a pointer that points to the internal buffer which contains the decimal value in packed decimal format, for example two digits per byte, the last right nibble contains the sign.

The expression (digitsOf() >> 1) + 1 identifies the number of bytes to store the decimal value.

Note: The buffer is not null terminated.

digitsOf

```
public:int digitsOf(void) const
```

This is supported on **z/OS**

Returns the number of digits represented by the IDecimalProxy value.

isNegative

```
public:inline int isNegative(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimalProxy object represents a negative value; otherwise returns zero (0).

isPositive

```
public:inline int isPositive(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimalProxy object represents a positive value; otherwise returns zero (0).

left

```
public:inline int left(void) const
```

This is supported on **z/OS**

Returns the value to the left of the decimal point, or width - precision.

precisionOf

```
public:int precisionOf(void) const
```

This is supported on **z/OS**

Returns the number of digits of precision.

right

```
public:inline int right(void) const
```

This is supported on **z/OS**

Returns the precision value of the IDecimalProxy object.

Type Conversion Functions

Use these functions to convert the decimal object from one type to another.

asBCD

```
public:IBinaryCodedDecimal asBCD(void) const
```

This is supported on **z/OS**

Returns the IDecimalProxy object as an IBinaryCodedDecimal representation.

asString

```
public:IString asString(void) const
```

This is supported on **z/OS**

Returns the IDecimalProxy object as an IString representation. The sign (+ or -) of the decimal is also included, and the number is zero filled to the specified DIGITS and PRECISION. If a sign was not specified when constructing, + sign is the default.

IDecimalProxy - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IDecimalBase			
Definition	**Page Number**	**Definition**	**Page Number**
AP	2861	APE	2862
APEC	2862	COMPEQ	2862

IDecimalBase			
Definition	**Page Number**	**Definition**	**Page Number**
COMPGE	2862	COMPGT	2862
COMPLE	2862	COMPLT	2862
CVB	2862	CVBS	2862
CVD	2862	CVDE	2862
DP	2862	DPE	2862
DPEC	2862	EDIT	2863
EDITMK	2863	MP	2863
MPE	2863	MPEC	2863
PACK	2863	proxyPB	2864
resultPB	2864	SP	2863
SPE	2863	SPEC	2863
SRP	2863	SRPE	2863
SRPR	2863	SRPRE	2863
UNPACK	2864	ZAP	2864
ZAPE	2864	ZAPEC	2864

Inherited Protected Data

None

IDecimalResult

This class is designed for internal use only and is not supported.

Class header file: idecimal.hpp

IDecimalResult - Hierarchy List

IDecimalBase

IDecimalResult

IDecimalResult - Member Functions and Data by Group

Constructors & Destructor

Use these functions to construct IDecimalResult objects.

IDecimalResult

Overload 1

```
public:IDecimalResult( const IDecimalProxy&,
                       unsigned int w,
                       int p )
```

This is supported on **z/OS**

Overload 2

```
public:inline IDecimalResult(const IDecimalResult&)
```

This is supported on **z/OS**

Overload 3

```
public:inline IDecimalResult(const IDecimalProxy&)
```

This is supported on **z/OS**

Overload 4

```
public:inline IDecimalResult(unsigned int w, int p)
```

This is supported on **z/OS**

Overload 5

```
public:inline IDecimalResult(const char*, unsigned int w, int p)
```

This is supported on **z/OS**

Data Members

precision

```
public:const int precision
```

This is supported on **z/OS**

width

```
public:const unsigned int width
```

This is supported on **z/OS**

Query Functions

Use these functions to obtain information about the IDecimalResult object.

cData

```
public:const char* cData(void) const
```

This is supported on **z/OS**

Returns a pointer that points to the internal buffer which contains the IDecimalResult value in packed decimal format, for example two digits per byte, the last right nibble contains the sign.

The expression (digitsOf() >> 1) + 1 identifies the number of bytes to store the decimal value.

Note: The buffer is not null terminated.

digitsOf

```
public:int digitsOf(void) const
```

This is supported on **z/OS**

Returns the number of digits represented by the IDecimalResult value.

isNegative

```
public:inline int isNegative(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimalResult object represents a negative value; otherwise returns zero (0).

isPositive

```
public:inline int isPositive(void) const
```

This is supported on **z/OS**

Returns one (1) if the IDecimalResult object represents a positive value; otherwise returns zero (0).

left

```
public:inline int left(void) const
```

This is supported on **z/OS**

Returns the value to the left of the decimal point, or width - precision.

precisionOf

```
public:int precisionOf(void) const
```

This is supported on **z/OS**

Returns the number of digits of precision.

right

```
public:inline int right(void) const
```

This is supported on **z/OS**

Returns the precision value of the IDecimalResult object.

Type Conversion Functions

Use these functions to convert the IDecimalResult object from one type to another.

asBCD

```
public:IBinaryCodedDecimal asBCD(void) const
```

This is supported on **z/OS**

Returns the IDecimalResult object as an IBinaryCodedDecimal representation.

asString

```
public:IString asString(void) const
```

This is supported on **z/OS**

Returns the IDecimalResult object as an IString representation. The sign (+ or -) of the decimal is also included, and the number is zero filled to the specified DIGITS and PRECISION. If a sign was not specified when constructing, + sign is the default.

operator IDecimalProxy

```
public:inline operator IDecimalProxy() const
```

This is supported on **z/OS**

IDecimalResult - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

IDecimalBase			
Definition	**Page Number**	**Definition**	**Page Number**
AP	2861	APE	2862
APEC	2862	COMPEQ	2862
COMPGE	2862	COMPGT	2862
COMPLE	2862	COMPLT	2862
CVB	2862	CVBS	2862
CVD	2862	CVDE	2862
DP	2862	DPE	2862
DPEC	2862	EDIT	2863
EDITMK	2863	MP	2863
MPE	2863	MPEC	2863
PACK	2863	proxyPB	2864
resultPB	2864	SP	2863
SPE	2863	SPEC	2863
SRP	2863	SRPE	2863
SRPR	2863	SRPRE	2863
UNPACK	2864	ZAP	2864
ZAPE	2864	ZAPEC	2864

Inherited Protected Data

None

IDecimalUtil

The IDecimalUtil class is used to perform operations on decimal numbers. This class is designed for internal use only and is not supported.

Class header file: idecimal.hpp

IDecimalUtil - Hierarchy List

IDecimalUtil

IDecimalUtil - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct IDecimalUtil objects.

IDecimalUtil

Constructs an IDecimalUtil object.

Overload 1

```
public:IDecimalUtil()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor.

Overload 2

```
public:IDecimalUtil(unsigned long auLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from an unsigned long.
auLong
The unsigned long to construct from.

Overload 3

```
public:IDecimalUtil(double aDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from a double.
aDouble
The double to construct from.

Overload 4

```
public:IDecimalUtil(unsigned int auInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from an unsigned int.
auInt The unsigned int to construct from.

Overload 5

```
public:IDecimalUtil(int anInt)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from an int.
anInt The int to construct from.

Overload 6

```
public:IDecimalUtil(const IBinaryCodedDecimal& bcd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Copy constructor.
bcd The object to copy from.

Overload 7

```
public:IDecimalUtil(long long aLongLong)
```

This is supported on **AIX OS/400 z/OS**

Construct from a long long.
aLongLong
The long long to construct from.

Overload 8

```
public:IDecimalUtil(unsigned long long auLongLong)
```

This is supported on **AIX OS/400 z/OS**

Construct from an unsigned long long.

auLongLong
The unsigned long long to construct from.

Overload 9

```
public:IDecimalUtil(long double alDouble)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from a long double.
alDouble
The long double to construct from.

Overload 10

```
public:IDecimalUtil(float afloat)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from a float.
aFloat The float to construct from.

Overload 11

```
public:IDecimalUtil(long aLong)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from a long.
aLong The long to construct from.

Conversion Operators

Use these operators convert the value to C fundamental type formats.

operator IBinaryCodedDecimal

```
public:operator IBinaryCodedDecimal() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicit conversion operator to pull out an IBinaryCodedDecimal object.

Equality Operators

Use these operators to compare decimal utility objects for equality and/or relative magnitude.

operator !=

```
public:int operator !=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for inequality.
dut The other object to compare to.

operator <

```
public:int operator <(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for relative magnitude.
dut The other object to compare to.

operator <=

```
public:int operator <=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for relative magnitude or equality.
dut The other object to compare to.

operator ==

```
public:int operator ==(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for equality.
dut The other object to compare to.

operator >

```
public:int operator >(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for relative magnitude.
dut The other object to compare to.

operator >=

```
public:int operator >=(const IDecimalUtil& dut) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Compares one decimal util object to another for relative magnitude or equality.
dut The other object to compare to.

IDecimalUtil - Associated Globals

operator *

```
IDecimalUtil
  operator *( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalUtil
  operator +( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
IDecimalUtil
  operator -( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
IDecimalUtil
  operator /( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& os,
               const IBinaryCodedDecimal& bcd )
```

Streams this object out to an output stream, formatted as text.

Note: This method is **enabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& os,
               const IDecimalUtil& dut )
```

Streams this object out to an output stream, formatted as text.

Note: This method is **disabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& os,
               const IBinaryCodedDecimal& bcd )
```

Streams this object out to an output stream, formatted as text.

Note: This method is **disabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& os,
               const IDecimalUtil& dut )
```

Streams this object out to an output stream, formatted as text.

Note: This method is **enabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream&
  operator >>( std::istream& is,
               IBinaryCodedDecimal& bcd )
```

Creates and IBinaryCodedDecimal object based on an object in the input stream.

Note: This method is **enabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream& operator >>(istream& is, IBinaryCodedDecimal& bcd)
```

Creates an IBinaryCodedDecimal based on the object in the input stream.

Note: This method is **disabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IDecimalUtil - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ILongLong

This class represents a 64-bit signed integer.

Class header file: ilonglng.hpp

ILongLong - Hierarchy List

ILongLong

ILongLong - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct and destruct ILongLong objects.

~ILongLong

```
public:~ILongLong()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

ILongLong

Constructs a long long object from a number of different types.

Overload 1

```
public:ILongLong( long mostSignificant,
          unsigned long leastSignificant )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from two long values (one signed and one unsigned) which represent the low and high portions of the long long.

Overload 2

```
public:ILongLong()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor which sets the value to zero.

Overload 3

```
public:ILongLong(double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from a double value. Any fractional part is ignored.

Overload 4

```
public:ILongLong(long double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from a long double. Any fractional part is ignored.

Overload 5

```
public:ILongLong(unsigned long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from an unsigned long value.

Overload 6

```
public:ILongLong(unsigned int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from an unsigned int value.

Overload 7

```
public:ILongLong(long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from a long value.

Overload 8

```
public:ILongLong(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs from an int.

Conversion Operators

Converts the object's value to other binary formats.

operator long double

```
public:operator long double() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Implicit conversion to obtain the value as a double.

Hash Support

Use this member function to obtain a hash value for this object.

Hash

```
public:HashResult Hash() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a hash value for this object.

Mathematical Operators

Use these operators to perform mathematical operations on long long objects.

```
public:ILongLong & operator| = ( ) (const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Does a bitwise OR of this object with the passed object

operator %=

```
public:ILongLong& operator %=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Modulo divides this object by the passed long long object.

operator &=

```
public:ILongLong& operator &=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Performs a bitwize AND operation on this object using the passed long long object.

operator *=

```
public:ILongLong& operator *=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Multiplies this object by the passed long long object.

operator ++

Pre-increments the value of this object.

Overload 1

```
public:ILongLong& operator ++()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ILongLong operator ++(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +=

```
public:ILongLong& operator +=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the passed long long object to this object.

operator --

Post-decrements the value of this object.

Overload 1

```
public:ILongLong operator --(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ILongLong& operator --()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -=

```
public:ILongLong& operator -=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subtracts the passed long long object from this object.

operator /=

```
public:ILongLong& operator /=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Divides this object by the passed long long object.

operator ^=

```
public:ILongLong& operator ^=(const ILongLong&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Does a bitwise XOR of this object with the passed object.

Member Access

Use these member functions to access member data of the long long object.

GetLeastSignificant

```
public:unsigned long GetLeastSignificant() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the least significant field of the long long.

GetMostSignificant

```
public:long GetMostSignificant() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the most significant field of this long long object.

Min-Max Support

Use these member functions to obtain the minimum and maximum values for long long objects.

ILongLongMax

```
public:static ILongLong ILongLongMax(void)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a long long which contains the maximum long long value.

ILongLongMin

```
public:static ILongLong ILongLongMin(void)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a long long which contains the minimum long long value.

Shift Operators

Use these member functions to shift the long long value to the left or right.

operator <<=

```
public:ILongLong& operator <<=(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Shifts this object left by the indicated number of bits.

operator >>=

```
public:ILongLong& operator >>=(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Shifts this object right by the indicated number of bits.

ILongLong - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPair

The IPair class provides generic ordered pairs of coordinates. The class serves as the base for the following specific ordered-pair classes:

- IPoint
- ISize
- IRange

This class provides basic utilities to get and set the two coordinate values. In addition, it provides a full set of comparison and mathematical operators to manipulate ordered pairs.

Class header file: ipoint.hpp

IPair - Hierarchy List

IPair

IPoint

IRange

ISize

IPair - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class. This class uses the compiler-generated copy constructor and assignment operator to copy and assign IPair objects.

IPair

Constructs an IPair object.

Overload 1

```
public:IPair(Coord coord1, Coord coord2)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct from two coordinate values.

Overload 2

```
public:IPair()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor, sets both values to zero.

Overload 3

```
public:IPair(Coord init)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Construct a pair object from a single coordinate, which is used for both pair values.

Comparison Operators

Use these operators to compare one IPair object to another.

operator !=

```
public:bool operator !=(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if either coordinate differs.

operator <

```
public:bool operator <(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if both coordinates are less than those of the specified **aPair**.

operator <=

```
public:bool operator <=(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if both coordinates are less than or equal.

operator ==

```
public:bool operator ==(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if both coordinates match those of the specified **aPair**.

operator >

```
public:bool operator >(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if both coordinates are greater than those of the specified **aPair**.

operator >=

```
public:bool operator >=(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

True if both coordinates are greater than or equal.

Conversions

Use these member functions to return an IPair object in a different form.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts the ordered pair to an IString containing a diagnostic representation of the object.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Converts the ordered pair (a, b) to an IString(″(a, b)″).

operator -

```
public:IPair operator -() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ordered pair whose coordinates are the difference between the corresponding coordinates of **pair1** and **pair2**.

When you use the unary format, it returns an ordered pair with negated coordinates.

Coordinates

Use these member functions to query and change the ordered pair of integers in an IPair object.

coord1

```
public:Coord coord1() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obtains the value of the first coordinate.

coord2

```
public:Coord coord2() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obtains the value of the second coordinate.

setCoord1

```
public:IPair& setCoord1(Coord coord1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the value of the first coordinate.

setCoord2

```
public:IPair& setCoord2(Coord coord2)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the value of the second coordinate.

Manipulation

Use these member function and operators to alter the coordinate values of an IPair object. This includes both member and nonmember arithmetic operators and members to scale the value of an IPair object.

operator %=

Replaces the coordinates with the remainder when divided by those of the following specified parameters:

aPair The library performs the remainder function between the corresponding coordinates, coord1 with coord1 of **aPair** and coord2 with coord2.

divisor
The library performs the remainder function between each coordinate and the **divisor**.

Overload 1

```
public:IPair& operator %=(ioc::numeric_t divisor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IPair& operator %=(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *=

Multiplies the coordinates by those of the specified parameter:

aPair The library performs the product function between the corresponding coordinates, coord1 with coord1 of **aPair** and coord2 with coord2.

multiplier
The library performs the product function between each coordinate and the **multiplier**.

Overload 1

```
public:IPair& operator *=(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IPair& operator *=(double multiplier)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +=

```
public:IPair& operator +=(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds the coordinates of the specified **aPair** to the coordinates of an ordered pair.

operator -=

```
public:IPair& operator -=(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Subtracts the coordinates specified in **aPair** from the IPair coordinates.

operator /=

Divides the coordinates by those of the second specified parameter:

aPair The library performs the quotient function between the corresponding coordinates, coord1 with coord1 of **aPair** and coord2 with coord2.

divisor
The library performs the product function between each coordinate and the **divisor**.

Overload 1

```
public:IPair& operator /=(double divisor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IPair& operator /=(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

scaleBy

```
public:IPair& scaleBy(double xFactor, double yFactor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales the X-coordinate by **xFactor**; the Y-coordinate by **yFactor**.

scaledBy

```
public:IPair scaledBy(double xFactor, double yFactor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as IPair::scaleBy, but returns a new IPair, leaving the original unmodified.

Minimum and Maximum

Use these member functions to determine the smaller or larger of two IPair objects.

maximum

```
public:IPair maximum(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ordered pair whose coordinates are the maximum of the corresponding coordinates of the IPair and the specified IPair.

minimum

```
public:IPair minimum(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns an ordered pair whose coordinates are the minimum of the corresponding coordinates of the IPair and the specified IPair.

Miscellaneous

These member functions are additional, unrelated members of the IPair class.

distanceFrom

```
public:double distanceFrom(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the distance from some other ordered pair.

dotProduct

```
public:ioc::numeric_t dotProduct(const IPair& aPair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the dot product with another ordered pair.

transpose

```
public:IPair& transpose()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Swaps the coordinates of the ordered pair. The friend version of this function returns a new pair with transposed coordinates.

IPair - Type Definitions

Coord

```
typedef ioc::numeric_t Coord
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Type of the coordinate values (long integer).

IPair - Associated Globals

operator %

```
IPair operator %(const IPair& pair1, const IPair& pair2)
```

Returns an ordered pair whose coordinates are the remainder of the corresponding coordinates of **pair1** and **pair2**.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IPair operator *(const IPair& pair1, const IPair& pair2)
```

Returns an ordered pair whose coordinates are the product of **pair1** and **pair2**.

The library performs the product function between the corresponding coordinates: coord1 of **pair1** with coord1 of **pair2** and coord2 with coord2.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IPair operator *(const IPair& pair1, double multiplier)
```

Returns an ordered pair whose coordinates are the product of **pair1** and **multiplier**.

The library performs the product function between each coordinate of **pair1** and the **multiplier**.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IPair operator +(const IPair& pair1, const IPair& pair2)
```

Returns an ordered pair whose coordinates are the sums of the corresponding coordinates of **pair1** and **pair2**.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
IPair operator -(const IPair& pair1, const IPair& pair2)
```

Returns an ordered pair whose coordinates are the difference between the corresponding coordinates of **pair1** and **pair2**.

When you use the unary format, it returns an ordered pair with negated coordinates.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
IPair operator /(const IPair& pair1, const IPair& pair2)
```

Returns an ordered pair whose coordinates are the quotient of the corresponding coordinates of **pair1** and **pair2** specified parameter:

The library performs the quotient function between the corresponding coordinates: coord1 of **pair1** with coord1 of **pair2** and coord2 with coord2.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
IPair operator /(const IPair& pair1, double divisor)
```

Returns an ordered pair whose coordinates are the quotient of the corresponding coordinates of **pair1** and **divisor**

The library performs the quotient function between each coordinate of **pair1** and the **divisor**.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IPair& aRectangle )
```

Writes ordered pairs to ostreams in the conventional manner.

Note: This method is **enabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IPair& aRectangle )
```

Writes ordered pairs to ostreams in the conventional manner.

Note: This method is **disabled** if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

transpose

```
IPair transpose(const IPair& aPair)
```

Swaps the coordinates of the ordered pair. The friend version of this function returns a new pair with transposed coordinates.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPair - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPoint

The IPoint class represent points in two-dimensional space. In addition to all the functions inherited from its base class, IPair, the IPoint class provides additional functions.

Class header file: ipoint.hpp

IPoint - Hierarchy List

IPair

IPoint

IPoint - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class. This class uses the compiler-generated copy constructor and assignment operator to copy and assign IPoint objects.

IPoint

Constructs an IPoint object.

Overload 1

```
public:IPoint(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IPoint object with the X- and Y-coordinates of the specified IPair object.

Overload 2

```
public:IPoint(Coord x, Coord y)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an IPoint object with the specified X- and Y-coordinates.

Overload 3

```
public:IPoint(const struct tagPOINT& ptl)
```

This is supported on **WIN**

Constructs an IPoint object from a POINT structure defined by the Windows SDK.

Overload 4

```
public:IPoint(const struct _POINTL& ptl)
```

This is supported on **WIN OS/2**

Constructs an IPoint object from a POINTL structure defined by the Windows SDK or the OS/2 Toolkit.

Overload 5

```
public:IPoint()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The X- and Y-coordinates of the resulting IPoint object are both initialized to 0.

Conversions

Use these member functions to return an IPoint object in a different form.

asPOINTL

```
public:struct _POINTL asPOINTL() const
```

This is supported on **WIN OS/2**

Converts the point into a POINTL structure defined by the Windows SDK or the OS/2 Toolkit.

Coordinates

Use these member functions to query and change the X- and Y-coordinates of an IPoint object.

setX

```
public:IPoint& setX(Coord X)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the point's X-coordinate.

setY

```
public:IPoint& setY(Coord Y)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the point's Y-coordinate.

x

```
public:Coord x() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the point's X-coordinate.

y

```
public:Coord y() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the point's Y-coordinate.

IPoint - Inherited Member Functions and Data

Inherited Public Functions

IPair			
Definition	**Page Number**	**Definition**	**Page Number**
asDebugInfo	2884	asString	2884
coord1	2884	coord2	2884
distanceFrom	2887	dotProduct	2887
IPair	2882	maximum	2886
minimum	2886	operator !=	2883
operator %=	2885	operator *=	2885
operator +=	2885	operator -	2884
operator -=	2885	operator /=	2886
operator <	2883	operator <=	2883
operator ==	2883	operator >	2883
operator >=	2883	scaleBy	2886
scaledBy	2886	setCoord1	2884
setCoord2	2884	transpose	2887

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IPointArray

The IPointArray class represents an array of IPoint objects.

Class header file: iptarray.hpp

IPointArray - Hierarchy List

IPointArray

IPointArray - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class.

~IPointArray

```
public:~IPointArray()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destroys an object of this class.

IPointArray

Constructs an IPointArray object.

Overload 1

```
public:IPointArray( ioc::unumeric_t dimension = 0,
                    const IPoint* array = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IPointArray(const IPointArray& pointArray)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this function to construct an IPointArray object from an existing IPointArray object.

operator =

```
public:IPointArray& operator =(const IPointArray& pointArray)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Use this operator to assign one IPointArray object to another.

Comparisons

Use these operators to compare two point arrays.

operator !=

```
public:bool operator !=(const IPointArray& pointArray) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the arrays are not the same length or the points are not identical or both.

operator ==

```
public:bool operator ==(const IPointArray& pointArray) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the arrays are the same length and have identical points.

Data Access

Use these member functions and operators to access attributes of objects of this class.

add

```
public:IPointArray& add(const IPoint& point)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Adds a point to the end of the array.

insert

```
public:IPointArray&
  insert( ioc::unumeric_t index,
          const IPoint& point )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts a point before the index specified.

operator []

Returns a reference to the point at the specified index.

Overload 1

```
public:IPoint& operator [](ioc::unumeric_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:const IPoint& operator [](ioc::unumeric_t index) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

remove

```
public:IPointArray& remove(ioc::unumeric_t index)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Removes a point at the specified index.

resize

```
public:IPointArray& resize(ioc::unumeric_t newsize)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Increases or decreases the size of the array. New points are initialized to 0,0.

reverse

```
public:IPointArray& reverse()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Reverses the elements in the array.

reversed

```
public:IPointArray reversed() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a copy of the point array with its elements reversed.

size

```
public:ioc::unumeric_t size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the dimension of the array.

IPointArray - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IRange

The IRange class represents a range of IPair::Coord values between a specified lower and upper bound (inclusive).

Class header file: ipoint.hpp

IRange - Hierarchy List

IPair

IRange

IRange - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class. This class uses the compiler-generated copy constructor and assignment operator to copy and assign IRange objects.

IRange

Constucts an IRange object.

Overload 1

```
public:IRange()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRange(const IPair& aPair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IRange(Coord lower, Coord upper)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Coordinates

Use these member functions to query and change the ordered pair of integers in an IRange object.

lowerBound

```
public:Coord lowerBound() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the lower bound of the range.

setLowerBound

```
public:IRange& setLowerBound(Coord lower)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the lower bound of the range.

setUpperBound

```
public:IRange& setUpperBound(Coord upper)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the upper bound of the range.

upperBound

```
public:Coord upperBound() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the upper bound of the range.

Testing

Use these member functions to test coordinate values.

includes

```
public:bool includes(Coord aValue) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the range contains the specified coordinate value.

IRange - Inherited Member Functions and Data

Inherited Public Functions

IPair			
Definition	**Page Number**	**Definition**	**Page Number**
asDebugInfo	2884	asString	2884
coord1	2884	coord2	2884
distanceFrom	2887	dotProduct	2887
IPair	2882	maximum	2886
minimum	2886	operator !=	2883
operator %=	2885	operator *=	2885
operator +=	2885	operator -	2884
operator -=	2885	operator /=	2886
operator <	2883	operator <=	2883
operator ==	2883	operator >	2883
operator >=	2883	scaleBy	2886
scaledBy	2886	setCoord1	2884
setCoord2	2884	transpose	2887

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

IRectangle

The IRectangle class represents a rectangular area defined by two points that form opposite corners of the rectangle. These two points are referred to as the minimum and maximum points.

IRectangle objects are usable independently of the coordinate system in use. The minimum, or origin, is defined as the point with the lowest coordinate values. Therefore, in a coordinate space where 0,0 is the top-left corner and increasing a point's coordinate value moves it to the right and down, the minimum point of an IRectangle will be the top-left corner and the maximum corner the bottom-right corner. Conversely, in a coordinate space where 0,0 is the bottom-left corner and increasing a point's coordinate value moves it to the right and up, the minimum corner of an IRectangle will be the bottom left, and the maximum corner the top right.

IRectangle provides some member functions that use the terms ″right,″ ″left,″ ″top,″ and ″bottom.″ These are synonyms for the functions defined in terms of minimum and maximum corners. The directional orientation for the right, left, top, and bottom functions is correct for a coordinate space where 0,0 is the lower-left corner. For other coordinate systems, left and bottom are the sides of the rectangle with the lowest coordinate value, and top and right the sides with the highest.

Mathematically, a rectangle includes all the points on the lines that intersect its minimum corner. It does not include the points that lie on its edges that do not intersect the origin. This is important when you are doing detailed graphics work. For example, a rectangle specified as having a minimum of 0,0 and maximum of 10,20 will include the points 0,0 though 0,19 but will not include 0,20. Similarly, the points 1,0 through 9,0 are contained, but 10,0 is not.

Various graphics and windowing classes, as well as their member functions, use rectangles.

Class header file: irect.hpp

IRectangle - Hierarchy List

IRectangle

IRectangle - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class.

Note: The Open Class Library constructs rectangles by taking the two points that are given (or implied) as opposite corners. The minimum point, or origin, is set to be the lesser of the two points. This ensures that internally the origin and corner points always are such that the origin is less than or equal to the corner.

IRectangle

Constructs an IRectangle object.

Overload 1

```
public:IRectangle( Coord point1X,
                   Coord point1Y,
                   Coord point2X,
                   Coord point2Y )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a rectangle from four values representing coordinates of the corners.

point1X
The X-coordinate of point 1.

point1Y
The Y-coordinate of point 1.

point2X
The X-coordinate of point 2.

point2Y
The Y-coordinate of point 2.

Overload 2

```
public:IRectangle(const IPoint& point1, const IPoint& point2)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a rectangle from two points at opposite corners.

Overload 3

```
public:IRectangle(const struct _RECTL& rectl)
```

This is supported on **WIN OS/2**

Constructs a rectangle from a RECTL structure defined by the Windows SDK or the OS/2 Toolkit.

Overload 4

```
public:IRectangle(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a rectangle with corners at 0,0 and the specified location. The lower of **pair** and 0,0 will be the origin of the rectangle.

Overload 5

```
public:IRectangle(const struct tagRECT& rect)
```

This is supported on **WIN**

Constructs a rectangle from a RECT structure defined by the Windows SDK.

Overload 6

```
public:IRectangle(Coord width, Coord height)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a rectangle of the specified size with one corner at 0,0. The size is specified as follows:
width The width of the rectangle.
height The height of the rectangle.

Overload 7

```
public:IRectangle(const IPoint& point, const ISize& size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a rectangle from a point and size. The maximum point is calculated by adding the width of the size to the X-coordinate of the given point and adding the height of the size to the Y-coordinate of the given point.

Overload 8

```
public:IRectangle()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default constructor. The two corners of the resulting rectangle are both at 0,0.

Comparisons

Use these operators to compare two rectangles for equality or inequality.

operator !=

```
public:bool operator !=(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the rectangles differ, true is returned.

operator ==

```
public:bool operator ==(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the two rectangles are identical, true is returned. Identity of rectangles means that the two defining points are the same.

Conversions

Use these member functions to convert a rectangle into various formats.

asDebugInfo

```
public:IString asDebugInfo() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Renders the rectangle as a diagnostic representation.

asRECTL

```
public:struct _RECTL asRECTL() const
```

This is supported on **WIN OS/2**

Converts the rectangle into a RECTL structure defined by the Windows SDK or the OS/2 Toolkit.

asString

```
public:IString asString() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Renders the rectangle as an IString("IRectangle(x1,y1,x2,y2)").

Dimensions

Use these member functions to obtain information about a rectangle's size.

area

```
public:Coord area() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the area of the rectangle. The area is determined by multiplying the width of the rectangle by the height. For example, a rectangle defined from IPoint(1,1), IPoint(10,20) would have an area of 9*19, or 171.

height

```
public:Coord height() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the height of the rectangle. The height is determined by subtracting the Y-coordinate of the minimum point from the Y-coordinate of the maximum point.

maxX

```
public:Coord maxX() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X-coordinate of the vertical line that is opposite the origin of the rectangle.

maxY

```
public:Coord maxY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Y-coordinate of the horizontal line opposite the origin.

minX

```
public:Coord minX() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X-coordinate of the vertical line that passes though the origin.

minY

```
public:Coord minY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Y-coordinate of the horizontal line that passes though the origin of the rectangle.

size

```
public:ISize size() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the ISize(width, height).

width

```
public:Coord width() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the width of the rectangle. The width is determined by subtracting the X-coordinate of the minimum point from the X-coordinate of the maximum point.

Implementation

These member functions are used internally to implement the class.

validate

```
protected:IRectangle& validate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Corrects an invalid rectangle after creation, expansion, or intersection.

Note: If the rectangle is invalid, the points for both the origin and the corner of the rectangle are reset to IPoint(0, 0).

Manipulation

Use these member functions to modify the rectangle, changing its size, proportions, or location.

centerAt

```
public:IRectangle& centerAt(const IPoint& point)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the rectangle so that its center is at the specified point.

centeredAt

```
public:IRectangle centeredAt(const IPoint& point) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as **centerAt**, but returns a new rectangle, leaving the original unmodified.

expandBy

Moves the corners of the rectangle outward from the center by the specified amount. The specified amount can be either a scalar (long integer) or a point.

Overload 1

```
public:IRectangle& expandBy(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle& expandBy(Coord coord)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

expandedBy

Same as **expandBy**, but returns a new rectangle, leaving the original unmodified.

Overload 1

```
public:IRectangle expandedBy(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle expandedBy(Coord coord) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

moveBy

```
public:IRectangle& moveBy(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the rectangle by the amount specified by **aPair**.

movedBy

```
public:IRectangle movedBy(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as **moveBy**, but returns a new rectangle, leaving the original unmodified.

movedTo

```
public:IRectangle movedTo(const IPoint& point) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as **moveTo**, but returns a new rectangle, leaving the original unmodified.

moveTo

```
public:IRectangle& moveTo(const IPoint& point)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the rectangle so that its origin corner is at the specified point.

scaleBy

Scales the rectangle by the specified amount. Scaling a rectangle multiplies its coordinates by the scale amount.

Overload 1

```
public:IRectangle& scaleBy(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by a point specifying the amounts in the X-axis and Y-axis directions.

Overload 2

```
public:IRectangle& scaleBy(double xfactor, double yfactor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by a pair of doubles. The function uses the first double to scale in the X-axis direction; the second in the Y-axis direction.

Overload 3

```
public:IRectangle& scaleBy(Coord coord)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by a long integer value.

Overload 4

```
public:IRectangle& scaleBy(double factor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by a double value.

scaledBy

Same as **scaleBy**, but returns a new rectangle, leaving the original unmodified.

Overload 1

```
public:IRectangle scaledBy(double factor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle scaledBy(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IRectangle scaledBy(Coord coord) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IRectangle scaledBy(double xfactor, double yfactor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

shrinkBy

Moves the corners of the rectangle inward toward the center by the specified amount, either a scalar or a point.

Note: shrinkBy(anAmount) is always equivalent to expandBy(-anAmount), and vice versa.

Overload 1

```
public:IRectangle& shrinkBy(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle& shrinkBy(Coord coord)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

shrunkBy

Same as **shrinkBy**, but returns a new rectangle, leaving the original unmodified.

Overload 1

```
public:IRectangle shrunkBy(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle shrunkBy(Coord coord) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

sizeBy

Scales the rectangle by the specified value, leaving the rectangle at the same location because the bottom-left point remains fixed.

Overload 1

```
public:IRectangle& sizeBy(double factor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by the same double factor in both the X-axis and Y-axis directions.

Overload 2

```
public:IRectangle& sizeBy(double xfactor, double yfactor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by two doubles specifying factors in the X-axis and Y-axis directions, respectively.

Overload 3

```
public:IRectangle& sizeBy(Coord factor)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by the same integer factor in both the X-axis and Y-axis directions.

Overload 4

```
public:IRectangle& sizeBy(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Scales by a pair of integer scalars specifying different factors in the X-axis and Y-axis directions.

sizedBy

Same as **sizeBy**, but returns a new rectangle, leaving the original unmodified.

Overload 1

```
public:IRectangle sizedBy(double factor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:IRectangle sizedBy(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 3

```
public:IRectangle sizedBy(Coord factor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 4

```
public:IRectangle sizedBy(double xfactor, double yfactor) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

sizedTo

```
public:IRectangle sizedTo(const IPair& pair) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Same as **sizeTo**, but returns a new rectangle, leaving the original unmodified.

sizeTo

```
public:IRectangle& sizeTo(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sizes the rectangle to the specified size.

Manipulation Operators

Use these operators to find a rectangle's union and intersection with another rectangle.

operator &

```
public:IRectangle operator &(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a rectangle representing the intersection of the specified rectangles.

operator &=

```
public:IRectangle& operator &=(const IRectangle& rectangle)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the rectangle to its intersection with the specified rectangle.

operator |

```
public:IRectangle operator |(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the rectangle representing the union of the specified rectangles. This is the smallest rectangle that encompasses both specified rectangles.

operator |=

```
public:IRectangle& operator |=(const IRectangle& rectangle)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Resets the rectangle to its union with the specified rectangle.

Points

Use these member functions to access points on or within the rectangle. You can query any of nine points on a rectangle's perimeter or its center by using these members.

centerXCenterY

```
public:IPoint centerXCenterY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center point of the rectangle.

centerXMaxY

```
public:IPoint centerXMaxY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center point of the horizontal line opposite the origin of the rectangle.

centerXMinY

```
public:IPoint centerXMinY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center point of the horizontal line passing through the origin of the rectangle.

maxXCenterY

```
public:IPoint maxXCenterY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center point of the vertical line opposite the origin of the rectangle.

maxXMaxY

```
public:IPoint maxXMaxY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the corner of the rectangle opposite the origin.

maxXMinY

```
public:IPoint maxXMinY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the corner of the rectangle at the other end of the horizontal line passing though the origin.

minXCenterY

```
public:IPoint minXCenterY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center of the vertical line that passes through the origin.

minXMaxY

```
public:IPoint minXMaxY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the corner of the rectangle at the other end of the vertical line passing though the origin.

minXMinY

```
public:IPoint minXMinY() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the origin corner of the rectangle.

Synonyms

Use these member functions when you are working in a coordinate space where 0,0 is the bottom-left corner. In this case the right-left-top-bottom orientation is correct. You can still use these members in other coordinate systems, but left and bottom are the sides of the rectangle with the lowest coordinate value, and top and right are the sides with the highest.

The following table lists the members and synonyms defined for them:

Function	Synonym
minXMinY	bottomLeft
minXCenterY	leftCenter
minXMaxY	topLeft
centerXMinY	bottomCenter
centerXCenterY	center
centerXMaxY	topCenter
maxXMinY	bottomRight
maxXCenterY	rightCenter
maxXMaxY	topRight
minX	left
minY	bottom
maxX	right
maxY	top

bottom

```
public:Coord bottom() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Y-coordinate of the horizontal line that forms the bottom of the rectangle. This is an alias for minY.

bottomCenter

```
public:IPoint bottomCenter() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the bottom-center point of the rectangle. This is an alias for centerXMinY.

bottomLeft

```
public:IPoint bottomLeft() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the bottom-left corner of the rectangle This is an alias for minXMinY.

bottomRight

```
public:IPoint bottomRight() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the bottom-right corner of the rectangle. This is an alias for maxXMinY.

center

```
public:IPoint center() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the center point of the rectangle. This is an alias for centerXCenterY.

left

```
public:Coord left() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X-coordinate of the vertical line that forms the left side of the rectangle. This is an alias for minX.

leftCenter

```
public:IPoint leftCenter() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the left-center point of the rectangle. This is an alias for minXCenterY.

right

```
public:Coord right() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X-coordinate of the vertical line that forms the right side of the rectangle. This is an alias for maxX.

rightCenter

```
public:IPoint rightCenter() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the right-center point of the rectangle. This is an alias for maxXCenterY.

top

```
public:Coord top() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the Y-coordinate of the horizontal line that forms the top of the rectangle. This is an alias for maxY.

topCenter

```
public:IPoint topCenter() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the top-center point of the rectangle. This is an alias for centerXMaxY.

topLeft

```
public:IPoint topLeft() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the top-left corner of the rectangle. This is an alias for minXMaxY.

topRight

```
public:IPoint topRight() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the X- and Y-coordinates of the top-right corner of the rectangle. This is an alias for maxXMaxY.

Testing

Use these member functions to test various attributes of a rectangle.

contains

If the rectangle contains the specified point or rectangle, true is returned. A point is contained by a rectangle if its coordinates are greater than or equal to the minimum point of the rectangle and less than the maximum point. A rectangle is contained within another rectangle if its minimum point is greater than or equal to the containing rectangle's minimum point and its maximum point is less than or equal to the containing rectangle's maximum point.

Overload 1

```
public:bool contains(const IPoint& point) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:bool contains(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

intersects

```
public:bool intersects(const IRectangle& rectangle) const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the rectangle and specified rectangle overlap, true is returned.

IRectangle - Type Definitions

Coord

```
typedef IPair::Coord Coord
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Type of the coordinate values; this must match the type of the coordinates supported by the IPair class.

IRectangle - Associated Globals

operator <<

```
std::ostream&
  operator <<( std::ostream& stream,
               const IRectangle& rectangle )
```

Streams this object out to an ostream.

Note: This method is enabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& stream,
               const IRectangle& rectangle )
```

Streams this object out to an ostream.

Note: This method is disabled if the macro __IOC_ANSI_STREAM is defined.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IRectangle - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

ISize

The ISize class uses its coordinates to represent a rectangular size, in horizontal and vertical dimensions.

Class header file: ipoint.hpp

ISize - Hierarchy List

IPair

ISize

ISize - Member Functions and Data by Group

Constructors & Destructor

Use these member functions to construct, copy, and assign objects of this class. This class uses the compiler-generated copy constructor and assignment operator to copy and assign ISize objects.

ISize

Constructs an ISize object.

Overload 1

```
public:ISize(const IPair& pair)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 2

```
public:ISize(Coord width, Coord height)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ISize object with the specified width and height.

Overload 3

```
public:ISize(const SIZEL& sizl)
```

This is supported on **WIN OS/2**

Constructs an ISize object from a SIZEL structure defined by the Windows SDK or the OS/2 Toolkit.

Overload 4

```
public:ISize()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Overload 5

```
public:ISize(const struct _RECTL& rcl)
```

This is supported on **WIN OS/2**

Constructs an ISize object from a RECTL structure defined by the Windows SDK or the OS/2 Toolkit. The object is initialized with the size of the RECTL structure.

Conversions

Use these member functions to return an ISize object in a different form.

asSIZEL

```
public:SIZEL asSIZEL() const
```

This is supported on **WIN OS/2**

Returns the ISize as a Presentation Manager Toolkit SIZEL structure.

Coordinates

Use these member functions to query and change the ordered pair of integers in an ISize object.

height

```
public:Coord height() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the height represented by the ISize object.

setHeight

```
public:ISize& setHeight(Coord cy)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the size's height.

setWidth

```
public:ISize& setWidth(Coord cx)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the size's width.

width

```
public:Coord width() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the width represented by the ISize object.

ISize - Inherited Member Functions and Data

Inherited Public Functions

IPair			
Definition	**Page Number**	**Definition**	**Page Number**
asDebugInfo	2884	asString	2884
coord1	2884	coord2	2884
distanceFrom	2887	dotProduct	2887
IPair	2882	maximum	2886
minimum	2886	operator !=	2883
operator %=	2885	operator *=	2885
operator +=	2885	operator -	2884
operator -=	2885	operator /=	2886
operator <	2883	operator <=	2883
operator ==	2883	operator >	2883
operator >=	2883	scaleBy	2886
scaledBy	2886	setCoord1	2884
setCoord2	2884	transpose	2887

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

Chapter 10. USL I/O Streaming Classes

_CCSID_T

The C++ Standard Library and the USL Library use this class to pass Coded Character Set ID (CCSID) information to the streaming functions. There are two identical versions of this class, one for the C++ Standard Library in the std namespace and the other for the USL Library in the global namespace.

Class header file: fstream.h

_CCSID_T - Hierarchy List

_CCSID_T

_CCSID_T - Member Functions and Data by Group

Constructors & Destructor

_CCSID_T

```
public:_CCSID_T(int ii)
```

This is supported on **OS/400**

Constructs an object of this class. The int parameter represents an OS/400 numeric CCSID.

Query Functions

value

```
public:int value() const
```

This is supported on **OS/400**

Returns the OS/400 numeric Coded Character Set IDentifier (CCSID).

_CCSID_T - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

filebuf

The filebuf class specializes streambuf for using files as the ultimate producer of the ultimate consumer.

In a filebuf object, characters are cleared out of the put area by doing write operations to the file, and characters are put into the get area by doing read

© Copyright IBM Corp. 2001

operations from that file. The filebuf class supports seek operations on files that allow seek operations. A filebuf object that is attached to a file descriptor is said to be open.

The stream buffer is allocated automatically if one is not specified explicitly with a constructor or a call to setbuf(). You can also create an unbuffered filebuf object by calling the constructor or setbuf() with the appropriate arguments. If the filebuf object is unbuffered, a system call is made for each character that is read or written.

The get and put pointers for a filebuf object behave as a single pointer. This single pointer is referred to as the get/put pointer. The file that is attached to the filebuf object also has a single pointer that indicates the current position where information is being read or written. This pointer is called the file get/put pointer.

Class header file: fstream.h

filebuf - Hierarchy List

streambuf

filebuf

filebuf - Member Functions and Data by Group

Constructors & Destructor

You can construct and destruct objects of the filebuf class.

~filebuf

```
public:~filebuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The filebuf destructor calls filebuf.close().

filebuf

Overload 1

```
public:filebuf(int fd, char* p, long l)
```

This is supported on **AIX**

Constructs a filebuf object that is attached to the file descriptor fd. The object is initialized to use the stream buffer starting at the position pointed to by p with length equal to l.

AIX Considerations

This function is available for 64-bit applications. The third argument is a long value.

Overload 2

```
public:filebuf(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a filebuf object that is attached to the file descriptor fd.

Overload 3

```
public:filebuf(int fd, char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a filebuf object that is attached to the file descriptor fd. The object is initialized to use the stream buffer starting at the position pointed to by p with length equal to l.

AIX Considerations

This function is available for 32-bit applications. The third argument is an int value.

Overload 4

```
public:filebuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an initially closed filebuf object.

Attach Functions

attach

Attaches the filebuf object to the file descriptor or the file pointer.

Overload 1

```
public:filebuf* attach(FILE* fp)
```

This is supported on **z/OS**

Attaches the filebuf object to the file pointer fp. If the filebuf object is already open, attach() returns 0. Otherwise, attach() returns a pointer to the filebuf object.

z/OS Considerations

If you have a file pointer already opened, use this function to do the attach instead of using the file descriptor.

Overload 2

```
public:filebuf* attach(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attaches the filebuf object to the file descriptor fd. If the filebuf object is already open or if fd is not open, attach() returns NULL. Otherwise, attach() returns a pointer to the filebuf object.

is_open

```
public:int is_open()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a nonzero value if the filebuf object is attached to a file descriptor. Otherwise, is_open() returns zero.

Data members

openprot

```
public:static const int openprot
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The default protection mode used when opening files.

in_start

```
protected:char* in_start
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member.

lahead

```
protected:char lahead [ 2 ]
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A variable used to store look-ahead characters during underflow processing.

last_seek

```
protected:streampos last_seek
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stream position last seeked to.

mode

```
protected:int mode
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Open mode of the filebuf object.

opened

```
protected:char opened
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A flag used to track whether the file is open. If the file is open, the value of this variable is 1. Otherwise it is 0.

xfd

```
protected:int xfd
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The file descriptor of the file attached to the filebuf object.

Detach Functions

close

```
public:filebuf* close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

close() does the following:

1. Flushes any output that is waiting in the filebuf object to be sent to the file
2. Disconnects the filebuf object from the file
3. Closes the file that was attached to the filebuf object.

If an error occurs, close() returns 0. Otherwise, close() returns a pointer to the filebuf object. Even if an error occurs, close() performs the second and third steps listed above.

detach

```
public:int detach()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Disconnects the filebuf object from the file without closing the file. If the filebuf object is not open, detach() returns -1. Otherwise, detach() flushes any output that is waiting in the filebuf object to be sent to the file, disconnects the filebuf object from the file, and returns the file descriptor.

File Pointer Functions

fp

```
public:FILE* fp()
```

This is supported on **z/OS**

Returns the file pointer that is attached to the filebuf object. If the filebuf object is not opened, fp() returns 0.

overflow

```
public:virtual int overflow(int = EOF)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Emptys an output buffer. Returns EOF when an error occurs. Returns 0 otherwise.

seekoff

```
public:virtual streampos seekoff(streamoff, ios::seek_dir, int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the file get/put pointer to the position specified by the ios::seek_dir argument with the offset specified by the streamoff argument. ios::seek_dir can have the following values:

- ios::beg - the beginning of the file
- ios::cur - the current position of the file get/put pointer
- ios::end - the end of the file

seekoff() changes the position of the file get/put pointer to the position specified by the value ios::seek_dir + streamoff. The offset can be either positive or negative. seekoff() ignores the third argument.

If the filebuf object is attached to a file that does not support seeking, or if the value of ios::seek_dir + streamoff specifies a position before the beginning of the file, seekoff() returns EOF and the position of the file get/put pointer is undefined. Otherwise, seekoff() returns the new position of the file get/put pointer.

z/OS Considerations

You can use relative byte offsets when seeking from ios::cur or ios::end. You can use relative byte offsets when seeking from ios::beg if either of the following conditions are true:

- The file is not a variable record format file, and is opened for binary I/O.
- The file is a variable record format file, and is opened for binary I/O with the byteseek option. The byteseek option is enabled for a specific file if the byteseek fopen() option is passed when the

file is opened. The byteseek option can also be enabled for all files if you set the _EDC_BYTESEEK environment variable.

When seeking from ios::beg in text files, encoded offsets are used. You can only seek to an offset value returned by a previous call to seekoff(), and attempting to calculate a new position based on an encoded offset value results in undefined behaviour.

sync

```
public:virtual int sync()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to synchronize the get/put pointer and the file get/put pointer. sync() may cause bytes that are waiting in the stream buffer to be written to the file, or it may reposition the file get/put pointer if characters that have been read from the file are waiting in the stream buffer. If it is not possible to synchronize the get/put pointer and the file get/put pointer, sync() returns EOF. If they can be synchronized, sync() returns zero.

underflow

```
public:virtual int underflow()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Fills an input buffer. Returns EOF when an error occurs or the end of the input is reached. Returns the next character otherwise.

Open Functions

z/OS Considerations

The prot parameter is ignored.

open

Opens the file with the name name and attaches the filebuf object to it. If name does not already exist and the open mode om does not equal ios::nocreate, open() tries to create it with protection mode equal to prot. The default value of prot is filebuf::openprot. An error occurs if the filebuf object is already open. If an error occurs, open() returns 0. Otherwise, open() returns a pointer to the filebuf object.

The default protection mode for the filebuf class is S_IREAD|S_IWRITE. If you create a file with both S_IREAD and S_IWRITE set, the file is created with both read and write permission. If you create a file with only S_IREAD set, the file is created with read-only permission, and cannot be deleted later with the stdio.h library function remove(). S_IREAD and S_IWRITE are defined in sys\stat.h.

Overload 1

```
public:filebuf*
  open( const char* name,
        int om,
        int prot = openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Overload 2

```
public:filebuf*
  open( const char* name,
        int om,
        int prot = openprot,
        _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Overload 3

```
public:filebuf*
  open( const char* name,
        const char* attr,
        int om,
        int prot = openprot )
```

This is supported on **z/OS**

You can use the attr parameter to specify additional file attributes, such as lrecl or recfm. All the parameters documented for the fopen() function are supported, with the exception of type=record.

Overload 4

```
public:filebuf* open(const char* name, int om, _CCSID_T ccsid)
```

This is supported on **OS/400**

Query Functions

fd

```
public:int fd()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the file descriptor that is attached to the filebuf object. If the filebuf object is closed, fd() returns EOF.

last_op

```
protected:int last_op()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Indicates whether the last operation was a read(get) or a write(put) operation.

Stream Buffer Functions

setbuf

Overload 1

```
public:virtual streambuf* setbuf(char* p, long len)
```

This is supported on **AIX**

Sets up a stream buffer with length in bytes equal to len, beginning at the position pointed to by p. setbuf() does the following:

- If p is 0 or len is nonpositive, setbuf() makes the filebuf object unbuffered.
- If the filebuf object is open and a stream buffer has been allocated, no changes are made to this stream buffer, and setbuf() returns NULL.

- If neither of these cases is true, setbuf() returns a pointer to the filebuf object.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value

Overload 2

```
public:virtual streambuf* setbuf(char* p, int len)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets up a stream buffer with length in bytes equal to len, beginning at the position pointed to by p. setbuf() does the following:

- If p is 0 or len is nonpositive, setbuf() makes the filebuf object unbuffered.
- If the filebuf object is open and a stream buffer has been allocated, no changes are made to this stream buffer, and setbuf() returns NULL.
- If neither of these cases is true, setbuf() returns a pointer to the filebuf object.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value

filebuf - Inherited Member Functions and Data

Inherited Public Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
~streambuf	3027	dbp	3031
in_avail	3028	optim_in_avail	3029
optim_sbumpc	3029	out_waiting	3036
overflow	3036	pbackfail	3036
pptr_non_null	3031	sbumpc	3029
seekoff	3032	seekpos	3032
setbuf	3038	sgetc	3029
sgetn	3029	snextc	3030
sputbackc	3037	sputc	3037
sputn	3037	stossc	3033
streambuf	3028	streambuf_resource	3038
xsgetn	3030	xsputn	3038

Inherited Public Data

None

Inherited Protected Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
allocate	3040	base	3033
blen	3041	doallocate	3041
eback	3033	ebuf	3033
egptr	3033	epptr	3033
gbump	3033	gptr	3034
pbase	3034	pbump	3034
pptr	3035	setb	3035
setg	3035	setp	3035
unbuffered	3041		

Inherited Protected Data

None

fstream

This class specializes the iostream class for use with files.

Class header file: fstream.h

fstream - Hierarchy List

ios

fstreambase

fstream

fstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the fstream class can be constructed and destructed.

~fstream

```
public:~fstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an fstream object.

fstream

Constructs an object of this class.

Overload 1

```
public:fstream(int fd, char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an fstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the fstream object. This constructor also sets up an associated filebuf object with a stream buffer that has length l bytes and begins at the

position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 32-bit applications. The third argument is an int value.

Overload 2

```
public:fstream(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Constructs an fstream object and opens the file name with open mode equal to mode and ccsid equal to ccsid.

If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 3

```
public:fstream(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an fstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the fstream object.

Overload 4

```
public:fstream(int fd, char* p, long l)
```

This is supported on **AIX**

Constructs an fstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the fstream object. This constructor also sets up an associated filebuf object with a stream buffer that has length l bytes and begins at the position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 64-bit applications. The third argument is a long value.

Overload 5

```
public:fstream( const char* name,
         int mode,
         int prot = filebuf::openprot,
         _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Constructs an fstream object and opens the file name with open mode equal to mode and protection mode equal to prot, and ccsid equal to ccsid.

The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 6

```
public:fstream( const char* name,
         const char* attr,
         int mode,
         int prot = filebuf::openprot )
```

This is supported on **z/OS**

Constructs an fstream object and opens the file name with open mode equal to mode, attributes equal to attr and protection mode equal to prot.

The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

You can use the attr parameter to specify additional file attributes such as lrecl or recfm. All the parameters documented for the fopen() functions are supported, with the exception of type=record.
z/OS Considerations

The prot attribute is ignored.

Overload 7

```
public:fstream( const char* name,
         int mode,
         int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Constructs an fstream object and opens the file name with open mode equal to mode and protection mode equal to prot.

The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.
z/OS Considerations

The prot attribute is ignored.

Overload 8

```
public:fstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an unopened fstream object.

Filebuf Functions

Use these functions to work with the underlying filebuf object.
rdbuf

```
public:filebuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the filebuf object that is attached to the fstream object.

Open Functions

Opens the file.

z/OS Considerations

The prot parameter is ignored.

open

Opens the specified file.

Overload 1

```
public:void
  open( const char* name,
        int mode,
        int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file of if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Overload 2

```
public:void
  open( const char* name,
        int mode,
        int prot = filebuf::openprot,
        _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Opens the file with the specified name, mode, protection and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file or if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 3

```
public:void
  open( const char* name,
        const char* attr,
        int mode,
        int prot = filebuf::openprot )
```

This is supported on **z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

You can use the attr parameter to specify additional file attributes, such as lrecl or recfm. All the parameters documented for the fopen() function are supported, with the exception of type=record.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Overload 4

```
public:void open(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Opens the file with the specified name, mode and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it unless ios::nocreate is set.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

fstream - Inherited Member Functions and Data

Inherited Public Functions

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~fstreambase	2927	attach	2930
close	2930	detach	2930
fstreambase	2927	setbuf	2932

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
verify	2930		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

fstreambase

The fstreambase class is an internal class that provides common functions for the classes that are derived from it; fstream, ifstream and ofstream. The fstreambase class inherits from the ios class. Do not use the fstreambase class directly.

Class header file: fstream.h

fstreambase - Hierarchy List

ios

fstreambase

ifstream

fstream

ofstream

fstreambase - Member Functions and Data by Group

Constructors & Destructor

Objects of the fstreambase class can be constructed and destructed by objects that derive from it. These constructors and destructors should not be used directly.

~fstreambase

```
public:~fstreambase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an fstreambase object.

fstreambase

Constructs an object of this class.

Overload 1

```
public:fstreambase(int fd, char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor does the following:

- constructs an fstreambase object
- initializes the filebuf object to the file descriptor passed in
- initializes the streambuf object and sets the get and put pointers based on the pointer **p** and the length **l**
- initializes the ios object.

If the file is already open, it clears the ios state. Otherwise, it sets the ios::failbit in the format state of the object.

AIX Considerations

This function is available for 32-bit applications. The third argument is an int value.

Overload 2

```
public:fstreambase( const char* name,
            const char* attr,
            int mode,
            int prot = filebuf::openprot )
```

This is supported on **z/OS**

Constructs an fstreambase object, initializes the ios object, and opens the specified file with the specified mode, attributes and protection.

You can use the attr parameter to specify additional file attributes such as lrecl or recfm. All the parameters documented for the fopen() functions are supported, with the exception of type=record.

z/OS Considerations

The prot parameter is ignored.

Overload 3

```
public:fstreambase(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor does the following:

- constructs an fstreambase object
- initializes the filebuf object to the file descriptor passed in
- initializes the ios object.

If the file is already open, it clears the ios state. Otherwise, it sets the ios::failbit in the format state of the object.

Overload 4

```
public:fstreambase(int fd, char* p, long l)
```

This is supported on **AIX**

This constructor does the following:

- constructs an fstreambase object
- initializes the filebuf object to the file descriptor passed in

- initializes the streambuf object and sets the get and put pointers based on the pointer **p** and the length **l**
- initializes the ios object.

If the file is already open, it clears the ios state. Otherwise, it sets the ios::failbit in the format state of the object.

AIX Considerations

This function is available for 64-bit applications. The third argument is a long value.

Overload 5

```
public:fstreambase( const char* name,
            int mode,
            int prot = filebuf::openprot,
            _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Constructs an fstreambase object, initializes the ios object, and opens the specified file with the specified mode, protection, and ccsid.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 6

```
public:fstreambase(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Constructs an fstreambase object, initializes the ios object, and opens the specified file with the specified mode and ccsid.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 7

```
public:fstreambase( const char* name,
            int mode,
            int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Constructs an fstreambase object, initializes the ios object, and opens the specified file with the specified mode and protection.

z/OS Considerations

The prot parameter is ignored.

Overload 8

```
public:fstreambase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Default constructor. Constructs an fstreambase object and initializes the ios object.

Filebuf Functions

Use these functions to work with the underlying filebuf object.

attach

Attaches the fstream, ifstream or ofstream object to the file descriptor or file pointer.

Overload 1

```
public:void attach(FILE* fp)
```

This is supported on **z/OS**

Attaches the fstream, ifstream or ofstream object to the file pointer fp. If the object is already attached to a file pointer, an error occurs and ios::failbit is set in the format state of the object.

Overload 2

```
public:void attach(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attaches the fstream, ifstream or ofstream object to the file descriptor fd. If the object is already attached to a file descriptor, an error occurs and ios::failbit is set in the format state of the object.

close

```
public:void close()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Closes the filebuf object, breaking the connection between the fstream, ifstream or ofstream object and the file descriptor. close() calls filebuf->close(). If this call fails, the error state of the fstream, ifstream or ofstream object is not cleared.

detach

```
public:int detach()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Detaches the filebuf object, breaking the connection between the fstream, ifstream or ofstream object and the file descriptor. detach() calls filebuf->detach().

rdbuf

```
public:filebuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the filebuf object that is attached to the fstream, ifstream or ofstream object.

Miscellaneous Functions

verify

```
protected:void verify(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Clears the format state of the object or sets the ios::failbit in the format state of the object depending on the value of the argument. If the argument value is 1, the format state is cleared, otherwise the ios::failbit is set.

Open Functions

open

Opens the specified file.

Overload 1

```
public:void
  open( const char* name,
        const char* attr,
        int mode,
        int prot = filebuf::openprot )
```

This is supported on **z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

You can use the attr parameter to specify additional file attributes, such as lrecl or recfm. All the parameters documented for the fopen() function are supported, with the exception of type=record.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Overload 2

```
public:void open(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Opens the file with the specified name, mode and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it unless ios::nocreate is set.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 3

```
public:void
  open( const char* name,
        int mode,
        int prot = filebuf::openprot,
        _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Opens the file with the specified name, mode, protection and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file or if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 4

```
public:void
  open( const char* name,
        int mode,
        int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file of if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Stream Buffer Functions

Use these functions to work with the underlying streambuf object.

setbuf

Overload 1

```
public:void setbuf(char* p, long l)
```

This is supported on **AIX**

Sets up a stream buffer with length in bytes equal to l beginning at the position pointed to by p. If p is equal to 0 or l is nonpositive, the fstream, ifstream or ofstream object (fb) will be unbuffered. If fb is open, or the call to fb->setbuf() fails, setbuf() sets ios::failbit in the object's state.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 2

```
public:void setbuf(char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets up a stream buffer with length in bytes equal to l beginning at the position pointed to by p. If p is equal to 0 or l is nonpositive, the fstream, ifstream or ofstream object (fb) will be unbuffered. If fb is open, or the call to fb->setbuf() fails, setbuf() sets ios::failbit in the object's state.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

fstreambase - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

ifstream

This class specializes the istream class for use with files.

Class header file: fstream.h

ifstream - Hierarchy List

ios

fstreambase

ifstream

ifstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the ifstream class can be constructed and destructed.

~ifstream

```
public:~ifstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ifstream object.

ifstream

Constructs an object of this class.

Overload 1

```
public:ifstream(int fd, char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ifstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ifstream object. This constructor also sets up an associated filebuf object with a stream buffer that has length l bytes and begins at the position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 32-bit applications. The third argument is an int value.

Overload 2

```
public:ifstream( const char* name,
          const char* attr,
          int mode = ios::in,
          int prot = filebuf::openprot )
```

This is supported on **z/OS**

Constructs an ifstream object and opens the file name with open mode equal to mode, attributes equal to attr and protection mode equal to prot. The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

You can use the attr parameter to specify additional file attributes such as lrecl or recfm. All the parameters documented for the fopen() functions are supported, with the exception of type=record.

z/OS Considerations

The prot attribute is ignored.

Overload 3

```
public:ifstream(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ifstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ifstream object.

Overload 4

```
public:ifstream(int fd, char* p, long l)
```

This is supported on **AIX**

Constructs an ifstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ifstream object. This constructor also sets up an associated filebuf

object with a stream buffer that has length l bytes and begins at the position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 64-bit applications. The third argument is a long value.

Overload 5

```
public:ifstream( const char* name,
          int mode = ios::in,
          int prot = filebuf::openprot,
          _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Constructs an ifstream object and opens the file name with open mode equal to mode and protection mode equal to prot, and ccsid equal to ccsid. The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 6

```
public:ifstream(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Constructs an ifstream object and opens the file name with open mode equal to mode and ccsid equal to ccsid. If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 7

```
public:ifstream( const char* name,
          int mode = ios::in,
          int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Constructs an ifstream object and opens the file name with open mode equal to mode and protection mode equal to prot. The default value for mode is ios::in and for prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed ifstream object is set.

z/OS Considerations

The prot attribute is ignored.

Overload 8

```
public:ifstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an unopened ifstream object.

Filebuf Functions

rdbuf

```
public:filebuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the filebuf object that is attached to the ifstream object.

Open Functions

Opens the file.

z/OS Considerations

The prot attribute is ignored.

open

Opens the specified file.

Overload 1

```
public:void
  open( const char* name,
        int mode = ios::in,
        int prot = filebuf::openprot,
        _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Opens the file with the specified name, mode, protection and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file or if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 2

```
public:void
  open( const char* name,
        int mode = ios::in,
        int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file of if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Overload 3

```
public:void open(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Opens the file with the specified name, mode and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it unless ios::nocreate is set.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 4

```
public:void
  open( const char* name,
        const char* attr,
        int mode = ios::in,
        int prot = filebuf::openprot )
```

This is supported on **z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

You can use the attr parameter to specify additional file attributes, such as lrecl or recfm. All the parameters documented for the fopen() function are supported, with the exception of type=record.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

ifstream - Inherited Member Functions and Data

Inherited Public Functions

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~fstreambase	2927	attach	2930
close	2930	detach	2930
fstreambase	2927	open	2931
setbuf	2932		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
verify	2930		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

ios

The ios class is the base class for the classes that format data that is extracted from or inserted into the stream buffer. The derived classes support the movement of formatted and unformatted data to and from the stream buffer.

The ios class maintains the format and error state information for the classes that are derived from it. The format state is a collection of flags and variables that can be set to control the details of formatting operations for input and output. The error state is a collection of flags that records whether any errors have taken place in the processing of the ios object. It also recoreds whether the end of an input stream has been reached.

Class header file: iostream.h

ios - Hierarchy List

ios

ostream

fstreambase

stdiostream

strstreambase

istream

ios - Member Functions and Data by Group

Constructors & Destructor

Objects of the ios class can be constructed and destructed.

~ios

```
public:virtual ~ios()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ios object.

ios

Creates an ios object.

Overload 1

```
public:ios(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The streambuf object is associated with the constructed ios object. If this argument is equal to 0, the result is undefined.

Overload 2

```
protected:ios()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This version of the ios constructor takes no arguments and is declared as protected. The ios class is used as a virtual base class for iostream, and therefore the ios class must have a constructor that takes no arguments. If you use this constructor in a derived class, you must use the init() function to associated the constructed ios object with the streambuf object.

Data members

adjustfield

```
public:static const long adjustfield
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

basefield

```
public:static const long basefield
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

floatfield

```
public:static const long floatfield
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

assign_private

```
protected:int assign_private
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

bp

```
protected:streambuf* bp
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class. Pointer to the streambuf object.

delbuf

```
protected:int delbuf
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

isfx_special

```
protected:int isfx_special
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

ispecial

```
protected:int ispecial
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

osfx_special

```
protected:int osfx_special
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

ospecial

```
protected:int ospecial
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

state

```
protected:int state
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

x_flags

```
protected:long x_flags
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

x_tie

```
protected:ostream* x_tie
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Data member for the ios class.

Error State Functions

bad

```
public:int bad() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a nonzero value is ios::badbit is set in the error state of the ios object. Otherwise, it returns 0.

ios::badbit is usually set when some operation on the streambuf object that is associated with the ios object has failed. It will probably not be possible to continue input and output operations on the ios object.

clear

```
public:void clear(int i = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Changes the error state of the ios object to the specified value. If the argument equals 0 (its default), all of the bits in the error state are cleared. If you want to set one of the bits without clearing or setting the other bits in the error state, you can perform a bitwise OR between the bit you want to set and the current error state. For example, the following statement sets ios::badbit in the ios object and leaves all the other error state bits unchanged:

```
iosobj.clear(ios::badbit | iosobj.rdstate());
```

eof

```
public:int eof() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a nonzero value if ios::eofbit is set in the error state of the ios object. Otherwise, it returns 0.

ios::eofbit is usually set when an EOF has been encountered during an extraction operation.

fail

```
public:int fail() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a nonzero value if either ios::badbit or ios::failbit is set in the error state. Otherwise, it returns 0.

good

```
public:int good() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a nonzero value if no bits are set in the error state of the ios object. Otherwise, it returns 0.

rdstate

```
public:int rdstate() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current value of the error state of the ios object.

setstate

```
protected:void setstate(int b)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Format State Functions

fill

Overload 1

```
public:char fill() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value of ios::x_fill of the ios object.

ios::x_fill is the character used as padding if the field is wider than the representation of a value. The default value for ios::x_fill is a space. The ios::left, ios::right and ios::internal flags determine the position of the fill character.

You can also use the parameterized manipulator setfill to set the value of ios::x_fill.

Overload 2

```
public:char fill(char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the value of ios::x_fill of the ios object to the specified character.

ios::x_fill is the character used as padding if the field is wider than the representation of a value. The default value for ios::x_fill is a space. The ios::left, ios::right and ios::internal flags determine the position of the fill character.

You can also use the parameterized manipulator setfill to set the value of ios::x_fill.

flags

Overload 1

```
public:long flags() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value of the flags that make up the current format state.

Overload 2

```
public:long flags(long f)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the flags in the format state to the settings specified in the argument and returns the value of the previous settings of the format flags.

precision

Overload 1

```
public:int precision() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value of ios::x_precision.

ios::x_precision controls the number of significant digits when floating-point values are inserted.

The format state in effect when precision() is called affects the behavior of precision(). If neither ios::scientific nor ios::fixed is set, ios::x_precision specifies the number of significant digits in the floating-point value that is being inserted. If, in addition, ios::showpoint is not set, all trailing zeros are removed and a decimal point only appears if it is followed by digits.

If either ios::scientific or ios::fixed is set, ios::x_precision specifies the number of digits following the decimal point.

Overload 2

```
public:int precision(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the value of ios::x_precision to the specified value and returns the previous value. The value must be greater than 0. If the value is negative, the value of ios::x_precision is set to the default value, 6.

You can also use the parameterized manipulator setprecision to set ios::x_precision.

The format state in effect when precision() is called affects the behavior of precision(). If neither ios::scientific nor ios::fixed is set, ios::x_precision specifies the number of significant digits in the floating-point value that is being inserted. If, in addition, ios::showpoint is not set, all trailing zeros are removed and a decimal point only appears if it is followed by digits.

If either ios::scientific or ios::fixed is set, ios::x_precision specifies the number of digits following the decimal point.

setf

Overload 1

```
public:long setf(long setbits, long field)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function clears the format flags specified in field, sets the format flags specified in setbits, and returns the previous value of the format state.

For example, to change the conversion base in the format state to ios::hex, you could use a statement like this:

```
s.setf(ios::hex, ios::basefield);
```

In this statement, ios::basefield specifies the conversion base as the format flag that is going to be changed and ios::hex specifies the new value for the conversion base. If setbits equals 0, all of the format flags specified in field are cleared.

You can also use the parameterized manipulator resetiosflags to clear format flags.

Note: If you set conflicting flags the results are unpredictable.

Overload 2

```
public:long setf(long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is accumulative. It sets the format flags that are specified in the argument, without affecting format flags that are not marked in the argument, and returns the previous value of the format state.

You can also use the parameterized manipulator setiosflags to set the format flags to a specific setting.

skip

```
public:int skip(int i)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the format flag ios::skipws if the value of the argument i does not equal 0. If i does equal 0, ios::skipws is cleared.

skip() returns a value of 1 if ios::skipws was set prior to the call to skip(), and returns 0 otherwise.

unsetf

```
public:long unsetf(long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Turns off the format flags specified in the argument and returns the previous format state.

width

Overload 1

```
public:int width() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value of the current setting of the format state field width variable, ios::x_width.

If the value of ios::x_width is smaller than the space needed for the representation of the value, the full value is still inserted.

Overload 2

```
public:int width(int w)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets ios::x_width to the value w and returns the previous value.

The default field width is 0. When the value of ios::x_width is 0, the operations that insert values only insert the characters needed to represent a value.

If the value of ios::x_width is greater than 0, the characters needed to represent the value are inserted. Then fill characters are inserted, if necessary, so that the representation of the value takes up the entire field. ios::x_width only specifies a minimum width, not a maximum width. If the number of characters needed to represent a value is greater than the field width, none of the characters is truncated. After every insertion of a value of a numeric or string type (including char*, unsigned char *, signed char*, and wchar_t*, but excluding char, unsigned char, signed char, and wchar_t), the value of ios::x_width is reset to 0. After every extraction of a value of type char*, unsigned char*, signed char*, or wchar_t*, the value of ios::x_width is reset to 0.

You can also use the parameterized manipulator setw to set the field width.

Format State Variables

The format state is a collection of format flags and format variables that control the details of formatting for input and output operations.

x_fill

```
protected:char x_fill
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Represents the character that is used to pad values that do not require the width of an entire field for their representation. Its default value is a space character.

x_precision

```
protected:short x_precision
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Represents the number of significant digits in the representation of floating-point values. Its default value is 6.

x_width

```
protected:short x_width
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Represents the minimum width of a field. Its default value is 0.

Initialization Functions

init

```
protected:void init(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Locking functions

ios_resource

```
public:IRTLResource& ios_resource() const
```

This is supported on **z/OS**

Miscellaneous Functions

bitalloc

```
public:static long bitalloc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

A static function that returns a long value with a previously unallocated bit set. You can use this long value as an additional flag, and pass it as an argument to the format state member functions. When all the bits are exhausted, bitalloc() returns 0.

iword

```
public:long& iword(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to the indexed user-defined flag, where the index used in the argument to this function is returned by xalloc().

iword() allocates space for the user-defined flag. If the allocation fails, iword() sets ios::failbit. You should check ios::failbit after calling iword().

operator !

```
public:int operator !() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The ! operator returns a nonzero value if ios::failbit or ios::badbit is set in the error state of the ios object.

For example, you could write:

```
if (!cin)
cout << "either ios::failbit or ios::badbit is set"  << endl;
else
cout << "neither ios::failbit nor ios::badbit is set" << endl;
```

operator const void *

```
public:operator const void *() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator void *

```
public:operator void *()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

pword

```
public:void *& pword(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a reference to a pointer to the indexed user-defined flag where the index used in the argument to this function is returned by xalloc().

pword() allocates space for the user-defined flag. If the allocation fails, pword() sets ios::failbit. You should check ios::failbit after calling pword().

On platforms where long and pointer types are the same size, pword() is the same as iword(), except that the two functions return different types.

rdbuf

```
public:streambuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the streambuf object that is associated with the ios object. This is the streambuf object that was passed as an argument to the ios constructor.

sync_with_stdio

```
public:static void sync_with_stdio()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

sync_with_stdio() is a static function that solves the problems that occur when you call functions declared in stdio.h and I/O Stream Library functions in the same program. The first time that you call sync_with_stdio(), it attaches stdiobuf objects to the predefined streams cin, cout and cerr. After that, input and output using these predefined streams can be mixed with input and output using the corresponding FILE objects (stdin, stdout, and stderr). This input and output are correctly synchronized.

If you switch between the I/O Stream Library formatted extraction functions and stdio.h functions, you may find that a byte is ″lost″. The reason is that the formatted extraction functions for integers and floating-point values keep extracting characters until a nondigit character is encountered. This nondigit character acts as a delimiter for the value that preceded it. Because it is not part of the value, putback() is called to return it to the stream buffer. If a C stdio library function, such as getchar(), performs the next input operation, it will begin input at the character after this nondigit character. Thus, this nondigit character is not part of the value extracted by the formatted extraction function, and it is not the character extracted by the C stdio library function. It is ″lost″. Therefore, you should avoid switching between the I/O Stream Library formatted extraction functions and C stdio library functions whenever possible.

sync_with_stdio() makes cout and clog unit buffered. After you call sync_with_stdio(), the performance of your program could diminish. The performance of your program depends on the length of strings, with performance diminishing most when the strings are shortest.

tie

Overload 1

```
public:ostream* tie()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the value of ios::x_tie.

ios::x_tie is the tie variable that points to the ostream object that is tied to the ios object.

You can use ios::x_tie to automatically flush the stream buffer attached to an ios object. If ios::x_tie for an ios object is not equal to 0 and the ios object needs more characters or has characters to be consumed, the ostream object pointed to by ios::x_tie is flushed.

By default, the tie variables of the predefined streams cin, cerr and clog all point to the predefined stream cout.

Overload 2

```
public:ostream* tie(ostream* s)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the tie variable, ios::x_tie, equal to the specified ostream and returns the previous value.

You can use ios::x_tie to automatically flush the stream buffer attached to an ios object. If ios::x_tie for an ios object is not equal to 0 and the ios object needs more characters or has characters to be consumed, the ostream object pointed to by ios::x_tie is flushed.

By default, the tie variables of the predefined streams cin, cerr and clog all point to the predefined stream cout.

xalloc

A static function that returns an unused index into an array of words available for use as format state variables by classes derived from ios.

xalloc() simply returns a new index; it does not do any allocation. iword() and pword() do the allocation, and if the allocation fails, they set ios::failbit. You should check ios::failbit after calling iword() or pword().

Overload 1

```
public:static int xalloc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

The value returned is an int for 32-bit applications. This function is not available for 64-bit applications.

Overload 2

```
public:static long xalloc()
```

This is supported on **AIX**
AIX Considerations

The value returned is a long for 64-bit applications. This function is not available for 32-bit applications.

(* stdioflush) ( )

```
protected:static void ( * stdioflush ) ( )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ios - Enumerations

White Space and Padding

The following values are set by default:

- skipws and right

skipws

If ios::skipws is set, white space will be skipped on input. If it is not set, white space is not skipped. If ios::skipws is not set, the arithmetic extractors will signal an error if you attempt to read an integer or floating-point value that is preceded by white space. ios::failbit is set, and

extraction ceases until it is cleared. This is done to avoid looping problems. If the following program is run with an input file that contains integer values separated by spaces, ios::failbit is set after the first integer value is read, and the program halts. If the program did not fail() at the beginning of the while loop to test if ios::failbit is set, it would loop indefinitely.

```
#include < fstream.h >
int main()
{
fstream f("spadina.dat", ios::in);
f.unsetf(ios::skipws);
int i;
while (!f.eof() && !f.fail()) {
f >> i;
cout << i;
}
}
```

left

If ios::left is set, the value is left-justified. Fill characters are added after the value.

right

If ios::right is set, the value is right-justified. Fill characters are added before the value.

internal

If ios::internal is set, the fill characters are added after any leading sign or base notation, but before the value itself.

Base Conversion

The manipulators ios::dec, ios::oct, and ios::hex have the same effect as the flags ios::dec, ios::oct, and ios::hex respectively. dec is set by default.

dec

If ios::dec is set, the conversion base is 10.

oct

If ios::oct is set, the conversion base is 8.

hex

If ios::hex is set, the conversion base is 16.

showbase

If ios::showbase is set, the operation that inserts values converts them to an external form that can be read according to the C++ lexical conventions for integral constants. By default, ios::showbase is unset.

Integral Formatting

The following manipulator affects integral formatting:

showpos

If ios::showpos is set, the operation that inserts values places a positive sign ″+″ into decimal conversions of positive integral values. By default, showpos is not set.

Floating-Point Formatting

The following format flags control the formatting of floating-point values:

showpoint

If ios:showpoint is set, trailing zeros and a decimal point appear in the result of a floating-point conversion. This flag has no effect if either ios::scientific or ios::fixed is set.

scientific

If ios::scientific is set, the value is converted using scientific notation. In scientific notation, there is one digit before the decimal point and the number of digits following the decimal point depends on the value of ios::x_precision. The default value for ios::x_precision is 6. If ios::uppercase is set, an uppercase ″E″ precedes the exponent. Otherwise, a lowercase ″e″ precedes the exponent.

fixed

If ios::fixed is set, floating point values are converted to fixed notation with the number of digits after the decimal point equal to the value of ios::x_precision (or 6 by default).

If neither ios::fixed nor ios::scientific is set, the representation of floating-point values depends on their values and the number of significant digits in the representation equals ios::x_precision. Floating-point values are converted to scientific notation if the exponent resulting from a conversion to scientific notation is less an -4 or greater than or equal to the value of ios::x_precision. Otherwise, floating-point values are converted to fixed notation. If ios::showpoint is not set, trailing zeros are removed from the result and a decimal point appears only if it is followed by a digit. ios::scientific and ios::fixed are collectively identified by the static member ios::floatfield.

Uppercase and Lowercase

uppercase

If ios::uppercase is set, the operation that inserts values uses an uppercase ″E″ for floating point values in scientific notation. In addition, the operation that inserts values stores hexadecimal digits ″A″ to ″F″ in uppercase and places an uppercase ″X″ before hexadecimal values when ios::showbase is set. If ios::uppercase is not set, a lowercase ″e″ introduces the exponent in floating-point values, hexadecimal digits ″a″ to ″f″ are stored in lowercase, and a lowercase ″x″ is inserted before hexadecimal values when ios::showbase is set.

Buffer Flushing

unitbuf

If ios::unitbuf is set, ostream::osfx() performs a flush after each insertion. The attached stream buffer is unit buffered.

stdio

This flag is used internally by sync_with_stdio(). You should not use ios::stdio directly.

Variation 1

```
enum  { skipws=01,
        left=02,
        right=04,
        internal=010,
        dec=020,
        oct=040,
        hex=0100,
        showbase=0200,
        showpoint=0400,
        uppercase=01000,
        showpos=02000,
        scientific=04000,
        fixed=010000,
        unitbuf=020000,
        stdio=040000 }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Variation 2

```
enum  { skipping=01000,
        tied=02000 }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

io_state

The error state state is an enumeration that records the errors that take place in the processing of ios objects.

Note: hardfail is a flag used internally by the I/O Stream Library. Do not use it.

open_mode

The elements of the open_mode enumeration have the following meanings:

- ios::app - open() performs a seek to the end of the file. Data that is written is appended to the end of the file. This value implies that the file is open for output.
- ios::ate - open() performs a seek to the end of the file. Setting ios::ate does not open the file for input or output. If you set ios::ate, you should explicitly set ios::in, ios::out, or both.
- ios::bin - See ios::binary below.
- ios::binary - The file is opened in binary mode. In the default (text) mode, carriage returns are discarded on input, as in an end-of-file (0x1a) character if it is the last character in the file. This means that a carriage return without an accompanying line feed causes the characters on either side of the carriage return to become adjacent. On output, a line feed is expanded to a carriage return and line feed. If you specify ios::binary, carriage returns and terminating end-of-file characters are not

removed on input, and a line feed is not expanded to a carriage return and line feed on output. ios::binary and ios::bin provide identical functionality.

- ios::in - The file is opened for input. If the file that is being opened for input does not exist, the open operation will fail. ios::noreplace is ignoredif ios::in is set.
- ios::out - The file is opened for output.
- ios::trunc - If the file already exists, its contents will be discarded. If you specify ios::out and neither ios::ate nor ios::app, you are implicitly specifying ios::trunc. If you set ios::trunc, you should explicitly set ios::in, ios::out, or both.
- ios::nocreate - If the file does not exist, the call to open() fails.
- ios::noreplace - If the file already exists and ios::out is et, the call to open() fails. If ios::out is not set, ios::noreplace is ignored.

Variation 1

```
enum open_mode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 bin=0200,
                 binary=bin }
```

This is supported on **WIN OS/2 AIX z/OS**

Variation 2

```
enum open_mode { in=1,
                 out=2,
                 ate=4,
                 app=010,
                 trunc=020,
                 nocreate=040,
                 noreplace=0100,
                 bin=0200,
                 binary=bin,
                 text=0400 }
```

This is supported on **OS/400**

seek_dir

The elements of the seek_dir enumeration have the following meanings:

- beg - the beginning of the ultimate producer or consumer
- cur - the current position in the ultimate producer or consumer
- end - the end of the ultimate producer or consumer

ios - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

iostream

This class combines the input capabilities of the istream class with the output capabilities of the ostream class. It is the base class for three other classes that also provide input and output capabilities:

- iostream_withassign - to assign another stream (such as an fstream for a file) to an iostream object.
- strstream - a stream of characters stored in memory.
- fstream - a stream that supports input and output.

Class header file: iostream.h

iostream - Hierarchy List

ios

istream

iostream

IFileUslIOStream

iostream_withassign

IFifoUslIOStream

iostream - Member Functions and Data by Group

Constructors & Destructor

Objects of the iostream class can be constructed and destructed.

~iostream

```
public:virtual ~iostream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an iostream object.

iostream

Overload 1

```
public:iostream(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes a single streambuf argument and creates an iostream object that is attached to the streambuf object. The constructor also initializes the format variables to their defaults.

Overload 2

```
protected:iostream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Protected contructor.

iostream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

istream			
Definition	**Page Number**	**Definition**	**Page Number**
~istream	2961	gcount	2962
get	2962	get_complicated	2966
getline	2966	ignore	2969
ipfx	2984	isfx	2985
istream	2961	operator >>	2972
peek	2970	putback	2983
read	2970	rs_complicated	2971
seekg	2983	sync	2984
tellg	2984		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

istream			
Definition	**Page Number**	**Definition**	**Page Number**
do_ipfx istream	2985 2961	eatwhite	2972

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( ) bp isfx_special osfx_special state x_flags x_tie	2950 2941 2942 2942 2942 2942 2942	assign_private delbuf ispecial ospecial x_fill x_precision x_width	2941 2941 2942 2942 2947 2947 2947

iostream_withassign

This class is derived from istream_withassign and ostream_withassign. Use this class to assign another stream to an iostream object.

Class header file: iostream.h

iostream_withassign - Hierarchy List

ios
  istream
    iostream
      iostream_withassign

iostream_withassign - Member Functions and Data by Group

Constructors & Destructor

~iostream_withassign

```
public:virtual ~iostream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an iostream_withassign object.

iostream_withassign

```
public:iostream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an iostream_withassign object. It does not do any initialization of this object.

operator =

```
public:iostream_withassign& operator =(iostream_withassign& rhs)
```

This is supported on **WIN OS/2 AIX**

Copy constructor.

Assignment Operators

operator =

Overload 1

```
public:iostream_withassign& operator =(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes a pointer to a streambuf object and associates this streambuf object with the iostream_withassign object that is on the left side of the assignment operator.

Overload 2

```
public:iostream_withassign& operator =(ios&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes a reference to an ios object and associates the stream buffer attached to this ios object with the iostream_withassign object that is on the left side of the assignment operator.

iostream_withassign - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945

ios			
Definition	**Page Number**	**Definition**	**Page Number**
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
~iostream	2955	iostream	2955

istream			
Definition	**Page Number**	**Definition**	**Page Number**
~istream	2961	gcount	2962
get	2962	get_complicated	2966
getline	2966	ignore	2969
ipfx	2984	isfx	2985
istream	2961	operator >>	2972
peek	2970	putback	2983
read	2970	rs_complicated	2971
seekg	2983	sync	2984
tellg	2984		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

istream			
Definition	**Page Number**	**Definition**	**Page Number**
do_ipfx	2985	eatwhite	2972
istream	2961		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

iostream			
Definition	**Page Number**	**Definition**	**Page Number**
iostream	2955		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

istream

You can use the istream class to perform formatted input, or extraction, from a stream buffer using the input operator >>. Consider the following statement, where ins is a reference to an istream object and x is a variable of a built-in type:

```
ins >> x;
```

The input operator >> calls ipfx(0). If ipfx() returns a nonzero value, the input operator extracts characters from the streambuf object that is associated with ins. It converts these characters to the type of x and stores the result x. The input operator sets ios::failbit if the characters extracted from the stream buffer cannot be converted to the type of x. If the attempt to extract characters fails because EOF is encountered, the input operator sets ios::eofbit and ios::failbit. If the attempt to extract characters fails for another reason, the input operator sets ios::badbit. Even if an error occurs, the input operator always returns ins.

The details of conversion depend on the format state of the istream object and the type of the variable x. The input operator may set the width variable ios::x_width to 0, but it does not change anything else in the format state.

The input operator is defined for the following types:

- Arrays of character values (including signed char and unsigned char)
- Other integral values: short, int, long, float, double, long double, and long long values.

In addition, the input operator is defined for streambuf objects.

You can also define input operators for your own types.

Class header file: iostream.h

istream - Hierarchy List

ios

istream

iostream

istream_withassign

istream - Member Functions and Data by Group

Constructors & Destructor

Objects of the istream class can be constructed and destructed.

~istream

```
public:virtual ~istream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an istream object.

istream

Overload 1

```
public:istream(streambuf*, int sk, ostream* t = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obsolete. Do not use.

Overload 2

```
public:istream(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes a single argument, a pointer to a streambuf, and creates an istream object that is attached to the streambuf object. The constructor also initializes the format variables to their defaults.

Note: The other istream constructor declarations in iostream.h are obsolete; do not use them.

Overload 3

```
public:istream(int size, char*, int sk = 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obsolete. Do not use.

Overload 4

```
public:istream(int fd, int sk = 1, ostream* t = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obsolete. Do not use.

Overload 5

```
protected:istream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Obsolete. Do not use.

Extract Functions

You can use the extract functions to extract characters from a stream buffer as a sequence of bytes. All of these functions call ipfx(1). They only proceed with their processing if ipfx(1) returns a nonzero value.

gcount

Returns the number of characters extracted from the stream buffer by the last call to an unformatted input function. The input operator >> may call unformatted input functions, and thus formatted input may affect the value returned by gcount().

Overload 1

```
public:int gcount()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

This function returns an int value for 32-bit applications. It is not available for 64-bit applications.

Overload 2

```
public:long gcount()
```

This is supported on **AIX**
AIX Considerations

This function returns a long value for 64-bit applications. It is not available for 32-bit applications.

get

Overload 1

```
public:int get()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a single character from the stream buffer attached to the istream object and returns it. Returns EOF if EOF is extracted. ios::failbit is never set.

Overload 2

```
public:istream& get(char*, int lim, char delim = '\n')
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.
- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 3

```
public:istream& get(unsigned char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a single character from the stream buffer attached to the istream object and stores this character in c.

Overload 4

```
public:istream& get(signed char* b, int lim, char delim = '\n')
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.
- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 5

```
public:istream& get(streambuf& sb, char delim = '\n')
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the streambuf, sb. The default value of the delim argument is '\n'. Extraction stops when any of the following conditions is true:

- an EOF character is encountered
- an attempt to store a character in sb fails

- ios::failbit is set in the error state of the istream object
- delim is encountered. delim is left in the stream buffer attached to the istream object.

Overload 6

```
public:istream& get(unsigned char* b, long lim, char delim = '\n')
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.
- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 7

```
public:istream& get(char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a single character from the stream buffer attached to the istream object and stores this character in c.

Overload 8

```
public:istream& get(signed char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a single character from the stream buffer attached to the istream object and stores this character in c.

Overload 9

```
public:istream& get(wchar_t&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts a single wchar_t character from the stream buffer attached to the istream object and stores this character in c.

Overload 10

```
public:istream& get(unsigned char* b, int lim, char delim = '\n')
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.
- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 11

```
public:istream& get(signed char* b, long lim, char delim = '\n')
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.
- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 12

```
public:istream& get(char*, long lim, char delim = '\n')
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. delim is left in the stream buffer and not stored in the array.

- lim - 1 characters are extracted without delim or EOF being encountered.

get() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

get_complicated

Overload 1

```
public:istream& get_complicated(signed char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 2

```
public:istream& get_complicated(unsigned char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 3

```
public:istream& get_complicated(char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

getline

Overload 1

```
public:istream&
  getline( unsigned char* b,
           int lim,
           char delim = '\n' )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 2

```
public:istream&
  getline( unsigned char* b,
           long lim,
           char delim = '\n' )
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 3

```
public:istream& getline(char* b, int lim, char delim = '\n')
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 4

```
public:istream& getline(char* b, long lim, char delim = '\n')
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 5

```
public:istream&
  getline( signed char* b,
           int lim,
           char delim = '\n' )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 6

```
public:istream&
  getline( signed char* b,
           long lim,
           char delim = '\n' )
```

This is supported on **AIX**

Extracts characters from the stream buffer attached to the istream object and stores them in the byte array beginning at the location pointed to by the first argument and extending for lim bytes. The default value of the delim argument is '\n'. Extraction stops when either of the following conditions is true:

- delim or EOF is encountered before lim - 1 characters have been stored in the array. getline() extracts delim from the stream buffer, but it does not store delim in the array.
- lim - 1 characters are extracted before delim or EOF is encountered.

getline() always stores a terminating null character in the array, even if it does not extract any characters from the stream buffer. ios::failbit is set if EOF is encountered before any characters are stored.

getline() is like get() with three arguments, except that get() does not extract the delim character from the stream buffer, while getline() does.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

ignore

```
public:istream& ignore(int n = 1, int delim = EOF)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts up to n characters from the stream buffer attached to the istream object and discards them. ignore() will extract fewer than n characters if it encounters delim or EOF.

peek

```
public:int peek()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

peek() calls ipfx(1). If ipfx() returns 0, or if no more input is available from the ultimate producer, peek() returns EOF. Otherwise, it returns the next character in the stream buffer without extracting the character.

read

Overload 1

```
public:istream& read(char* s, long n)
```

This is supported on **AIX**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 2

```
public:istream& read(signed char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 3

```
public:istream& read(unsigned char* s, long n)
```

This is supported on **AIX**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 4

```
public:istream& read(unsigned char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 5

```
public:istream& read(signed char* s, long n)
```

This is supported on **AIX**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 6

```
public:istream& read(char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts n characters from the stream buffer attached to the istream object and stores them in an array beginning at the position pointed to by s. If EOF is encountered before read() extracts n characters, read() sets the ios::failbit in the error state of the istream object. You can determine the number of characters that read() extracted by calling gcount() immediately after the call to read().
AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

rs_complicated

Overload 1

```
public:istream& rs_complicated(signed char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 2

```
public:istream& rs_complicated(char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 3

```
public:istream& rs_complicated(unsigned char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

eatwhite

```
protected:void eatwhite()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Input Operators

Input operators supported by istream objects.

operator >>

Overload 1

```
public:istream& operator >>(float&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator converts characters from the stream buffer attached to the input stream according to the C++ lexical conventions.

The following conversions occur for certain string values:

- If the value consists of the character strings ″inf″ or ″infinity″ in any combination of uppercase and lowercase letters, the string is converted to the approprate type's representation of infinity.
- If the value consists of the character string ″nan″ in any combination of uppercase and lowercase letters, the string is converted to the appropriate type's representation of a NaN.

Note that if you use thse string values as input in a program compiled with z/OS C/C++, the input operator will not recognize them as floating point numbers and will set ios::badbit in the stream's error state.

The resulting value is stored in the reference location provided. The input operator sets ios::failbit if no digits are available in the stream buffer or if the digits that are available do not begin a floating-point number.

Overload 2

```
public:istream& operator >>(char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator stores characters from the stream buffer attached to the input stream in the array pointed to by the argument. The input operator stores characters until a white-space character is found. This white-space character is left in the stream buffer, and the extraction stops. If ios::x_width does not equal 0, a maximum of ios::x_width - 1 characters are extracted. The input operator calls width(0) to reset the ios::x_width to 0.

The input operator always stores a terminating null character in the array, even if an error occurs.

Overload 3

```
public:istream& operator >>(int&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 4

```
public:istream& operator >>(long double&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator converts characters from the stream buffer attached to the input stream according to the C++ lexical conventions.

The following conversions occur for certain string values:

- If the value consists of the character strings ″inf″ or ″infinity″ in any combination of uppercase and lowercase letters, the string is converted to the approprate type's representation of infinity.
- If the value consists of the character string ″nan″ in any combination of uppercase and lowercase letters, the string is converted to the appropriate type's representation of a NaN.

Note that if you use thse string values as input in a program compiled with z/OS C/C++, the input operator will not recognize them as floating point numbers and will set ios::badbit in the stream's error state.

The resulting value is stored in the reference location provided. The input operator sets ios::failbit if no digits are available in the stream buffer or if the digits that are available do not begin a floating-point number.

Overload 5

```
public:istream& operator >>(ios & ( * f ) ( ios & ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The following built-in manipulators are accepted by this input operator:

```
ios&   dec(ios&)
ios&   hex(ios&)
ios&   oct(ios &)
```

These manipulators have a specific effect on an istream object beyond extracting their own values. For example, If ins is a reference to an istream object, then this statement sets ios::dec:

```
ins >> dec;
```

Overload 6

```
public:istream& operator >>(long long&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit

character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Note: The support for long long is controlled by _LONG_LONG, __EXTENDED__, or the -q(no)longlong option.

Overload 7

```
public:istream& operator >>(long&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character

that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.

- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 8

```
public:istream& operator >>(short&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter

from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 9

```
public:istream& operator >>(signed char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts a character from the stream buffer attached to the input stream and stores it in c.

Overload 10

```
public:istream& operator >>(signed char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator stores characters from the stream buffer attached to the input stream in the array pointed to by the argument. The input operator stores characters until a white-space character is found. This white-space character is left in the stream buffer, and the extraction stops. If ios::x_width does not equal 0, a maximum of ios::x_width - 1 characters are extracted. The input operator calls width(0) to reset the ios::x_width to 0.

The input operator always stores a terminating null character in the array, even if an error occurs.

Overload 11

```
public:istream& operator >>(unsigned char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator stores characters from the stream buffer attached to the input stream in the array pointed to by the argument. The input operator stores characters until a white-space character is found. This white-space character is left in the stream buffer, and the extraction stops. If ios::x_width does not equal 0, a maximum of ios::x_width - 1 characters are extracted. The input operator calls width(0) to reset the ios::x_width to 0.

The input operator always stores a terminating null character in the array, even if an error occurs.

Overload 12

```
public:istream& operator >>(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

For pointers to streambuf objects, the input operator calls ipfx(0). If ipfx(0) returns a nonzero value, the input operator extracts characters from the stream buffer attached to the istream object and inserts them in the streambuf. Extraction stops when an EOF character is encountered.

The input operator always returns a reference to the istream object.

Overload 13

```
public:istream& operator >>(unsigned int&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.

- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 14

```
public:istream& operator >>(unsigned long long&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Note: The support for long long is controlled by _LONG_LONG, __EXTENDED__, or the -q(no)longlong option.

Overload 15

```
public:istream& operator >>(unsigned long&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 16

```
public:istream& operator >>(unsigned short&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts characters from the stream buffer associated with the input stream and converts them according to the format state of the input stream. The converted characters are then store in the reference location provided. There is no overflow detection on conversion of integral types.

The first character extracted from the stream buffer may be a sign (+ or -). The subsequent characters are converted until a nondigit character is encountered. This nondigit character is left in the stream buffer. Which characters are treated as digits depends on the setting of the following format flags:

- ios::oct - the characters are converted to an octal value. Characters are extracted from the stream buffer until a character that is not an octal digit (a digit from 0 to 7) is encountered. If ios::oct is set and a signed value is encountered, the value is converted into a decimal value.
- ios::dec - the characters are converted to a decimal value. Characters are extracted from the stream buffer until a character that is not a decimal digit (a digit from 0 to 9) is encountered.
- ios::hex - the characters are converted to an hexadecimal value. Characters are extracted from the stream buffer until a character that is not a hexadecimal digit (a digit from 0 to 0 or a letter from ″A″ to ″F″, upper or lower case) is encountered. If ios::hex is set and a signed value is encountered, the value is converted into a decimal value.

If none of these format flags is set, the characters are converted according to the C++ lexical conventions. This conversion depends on the characters that follow the optional sign:

- If these characters are ″0x″ or ″0X″, the subsequent characters are converted to a hexadecimal value.
- If the first character is ″0″ and the second character is not a ″x″ or ″X″, the subsequent characters are converted to an octal value.
- If neither of these cases is true, the characters are converted to a decimal value.

If no digits are available in the stream buffer (other than the ″0″ in ″0X″ or ″0x″ preceding a hexadecimal value), the input operator sets ios::failbit in the error state of the input stream.

Overload 17

```
public:istream& operator >>(wchar_t&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts a wchar_t character from the stream buffer attached to the input stream and stores it in the reference location provided. If ios::skipws is set, the input operator skips leading wchar_t spaces as well as leading char white spaces.

Overload 18

```
public:istream& operator >>(wchar_t*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator stores characters from the stream buffer attached to the input stream in the array pointed to by the argument. The input operator stores characters until a white-space character or a wchar_t blank is found. If the terminating character

is a white-space character, it is left in the stream buffer. If it is a wchar_t blank, it is discarded to avoid returning two bytes to the input stream.

For wchar_t* arrays, if ios::x_width does not equal 0, a maximum of ios::x_width - 1 characters (at 2 bytes each) are extracted. A 2-character space is reserved for the wchar_t terminating null character.

The input operator resets ios::x_width to 0.

The input operator always stores a terminating null character in the array, even if an error occurs. For arrays of wchar_t*, this terminating null character is a wchar_t terminating null character.

Overload 19

```
public:istream& operator >>(unsigned char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts a character from the stream buffer attached to the input stream and stores it in c.

Overload 20

```
public:istream& operator >>(istream & ( * f ) ( istream & ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The following built-in manipulators are accepted by this input operator:

```
istream&   ws(istream&)
```

These manipulators have a specific effect on an istream object beyond extracting their own values. For example, If ins is a reference to an istream object, then this statement extracts white-space characters from the stream buffer attached to ins:

```
ins >> ws;
```

Overload 21

```
public:istream& operator >>(double&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator converts characters from the stream buffer attached to the input stream according to the C++ lexical conventions.

The following conversions occur for certain string values:

- If the value consists of the character strings ″inf″ or ″infinity″ in any combination of uppercase and lowercase letters, the string is converted to the approprate type's representation of infinity.
- If the value consists of the character string ″nan″ in any combination of uppercase and lowercase letters, the string is converted to the appropriate type's representation of a NaN.

Note that if you use thse string values as input in a program compiled with z/OS C/C++, the input operator will not recognize them as floating point numbers and will set ios::badbit in the stream's error state.

The resulting value is stored in the reference location provided. The input operator sets ios::failbit if no digits are available in the stream buffer or if the digits that are available do not begin a floating-point number.

Overload 22

```
public:istream& operator >>(char& c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The input operator extracts a character from the stream buffer attached to the input stream and stores it in c.

Positioning Functions

Functions that work with the get pointer of the ultimate producer.

putback

```
public:istream& putback(char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

putback() attempts to put an extracted character back into the stream buffer. c must equal the character before the get pointer of the stream buffer. Unless some other activity is modifying the stream buffer, this is the last character extracted from the stream buffer. If c is not equal to the character before the get pointer, the result of putback() is undefined, and the error state of the input stream may be set. putback() does not call ipfx(), but if the error state of the input stream is nonzero, putback() returns without putting back the character or setting the error state.

seekg

Overload 1

```
public:istream& seekg(streampos p)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the get pointer to the position p.

If you attempt to set the get pointer to a position that is not valid, seekg() sets ios::badbit.

Overload 2

```
public:istream& seekg(streamoff o, ios::seek_dir d)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the get pointer to the position specified by d with the offset o. The argument d can have the following values:

- ios::beg - the beginning of the stream
- ios::cur - the current position of the get pointer
- ios::end - the end of the stream

If you attempt to set the get pointer to a position that is not valid, seekg() sets ios::badbit.

sync

```
public:int sync()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Establishes consistency between the ultimate producer and the stream buffer attached to the input stream. sync() calls rdbuf()->sync(), which is a virtual function, so the details of its operation depend on the way the function is defined in a given derived class. If an error occurs, sync() returns EOF.

tellg

```
public:streampos tellg()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current position of the get pointer of the ultimate producer.

Prefix and Suffix Functions

Functions that are called either before or after extracting characters from the ultimate producer.

ipfx

Checks the stream buffer attached to an istream object to determine if it is capable of satisfying requests for characters. It returns a nonzero value if the stream buffer is ready, and 0 if it is not.

The formatted input operator calls ipfx(0), while the unformatted input functions call ipfx(1).

If the error state of the istream object is nonzero, ipfx() returns 0. Otherwise, the stream buffer attached to the istream object is flushed if either of the following conditions is true:

- noskipws has a value of 0. The number of characters available in the stream buffer is fewer than the value of noskipws.

If ios::skipws is set in the format state of the istream object and noskipws has a value of 0, leading white-space characters are extracted from the stream buffer and discarded. If ios::hardfail is set or EOF is encountered, ipfx() returns 0. Otherwise, it returns a nonzero value.

Overload 1

```
public:int ipfx(int noskipws = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

This function accepts an int value for 32-bit applications. It is not available for 64-bit applications.

Overload 2

```
public:int ipfx(long noskipws = 0)
```

This is supported on **AIX**
AIX Considerations

This function accepts a long value for 64-bit applications. It is not available for 32-bit applications.

isfx

```
public:void isfx()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

do_ipfx

Overload 1

```
protected:int do_ipfx(long noskipws)
```

This is supported on **AIX**

Internal function. Do not use.

AIX Considerations

This function is available for 64-bit applications. It accepts a long argument.

Overload 2

```
protected:int do_ipfx(int noskipws)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

istream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

istream_withassign

Use this class to assign another stream to an istream object.

Class header file: iostream.h

istream_withassign - Hierarchy List

ios

istream

istream_withassign

istream_withassign - Member Functions and Data by Group

Constructors & Destructor

Objects of the istream_withassign class can be constructed and destructed. They can also be copied.

~istream_withassign

```
public:virtual ~istream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ostream_withassign object.

istream_withassign

```
public:istream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an istream_withassign object. It does not do any initialization of this object.

operator =

```
public:istream_withassign& operator =(istream_withassign& rhs)
```

This is supported on **WIN OS/2 AIX**

The copy constructor.

Assignment Operator

Assignment operators for istream_withassign.

operator =

Overload 1

```
public:istream_withassign& operator =(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes a pointer to a streambuf object as its argument. It associates this streambuf object with the istream_withassign object that is on the left side of the assignment operator.

Overload 2

```
public:istream_withassign& operator =(istream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes an istream objects as its argument. It associates the stream buffer attached to the input stream with the istream_withassign object that is on the left side of the assignment operator.

istream_withassign - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944

ios			
Definition	**Page Number**	**Definition**	**Page Number**
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

istream			
Definition	**Page Number**	**Definition**	**Page Number**
~istream	2961	gcount	2962
get	2962	get_complicated	2966
getline	2966	ignore	2969
ipfx	2984	isfx	2985
istream	2961	operator >>	2972
peek	2970	putback	2983
read	2970	rs_complicated	2971
seekg	2983	sync	2984
tellg	2984		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

istream			
Definition	**Page Number**	**Definition**	**Page Number**
do_ipfx	2985	eatwhite	2972
istream	2961		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
`( * stdioflush ) ( )`	2950	`assign_private`	2941
`bp`	2941	`delbuf`	2941
`isfx_special`	2942	`ispecial`	2942
`osfx_special`	2942	`ospecial`	2942
`state`	2942	`x_fill`	2947
`x_flags`	2942	`x_precision`	2947
`x_tie`	2942	`x_width`	2947

istrstream

istrstream is the class that specializes istream to use a strstreambuf for extraction from arrays of characters in memory. You can create an istrstream object by associating the object with a previously allocated array of characters. You can then read input from it and apply other operations to it just as you would to another type of stream.

Class header file: strstream.h

istrstream - Hierarchy List

ios

strstreambase

istrstream

istrstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the istrstream class can be constructed and destructed.

~istrstream

```
public:~istrstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The istrstream destructor frees space that was allocated by the istrstream constructor.

istrstream

Overload 1

```
public:istrstream(const char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

Overload 2

```
public:istrstream(const signed char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

Overload 3

```
public:istrstream(char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 4

```
public:istrstream(signed char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 5

```
public:istrstream(const signed char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 6

```
public:istrstream(const signed char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 7

```
public:istrstream(const unsigned char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

Overload 8

```
public:istrstream(const unsigned char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 9

```
public:istrstream(const unsigned char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 10

```
public:istrstream(const char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 11

```
public:istrstream(signed char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

Overload 12

```
public:istrstream(unsigned char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

Overload 13

```
public:istrstream(unsigned char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 14

```
public:istrstream(unsigned char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 15

```
public:istrstream(signed char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 16

```
public:istrstream(const char* str, long size)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 17

```
public:istrstream(char* str, int size)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the array of bytes that starts at the position pointed to by str and has a length of size bytes. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 18

```
public:istrstream(char* str)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted from the null-terminated string that is pointed to by str. You can use the istream::seekg() function to reposition the get pointer in this string.

istrstream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
rdbuf	3047		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield floatfield	2941 2941	basefield	2941

Inherited Protected Functions

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~strstreambase	3046	strstreambase	3047

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

ofstream

This class specializes the ostream class for use with files.

Class header file: fstream.h

ofstream - Hierarchy List

ios

fstreambase

ofstream

ofstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the ofstream class can be constructed and destructed.

~ofstream

```
public:~ofstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ofstream object.

ofstream

Constructs an object of this class.

Overload 1

```
public:ofstream(int fd, char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ofstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ofstream object. This constructor also sets up an associated filebuf object with a stream buffer that has length l bytes and begins at the position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 32-bit applications. The third argument is an int value.

Overload 2

```
public:ofstream( const char* name,
          const char* attr,
          int mode = ios::out,
          int prot = filebuf::openprot )
```

This is supported on **z/OS**

Constructs an ofstream object and opens the file name with open mode equal to mode, attributes equal to attr and protection mode equal to prot. The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

You can use the attr parameter to specify additional file attributes such as lrecl or recfm. All the parameters documented for the fopen() functions are supported, with the exception of type=record.

z/OS Considerations

The prot attribute is ignored.

Overload 3

```
public:ofstream(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ofstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ofstream object.

Overload 4

```
public:ofstream(int fd, char* p, long l)
```

This is supported on **AIX**

Constructs an ofstream object that is attached to the file descriptor fd. If fd is not open, ios::failbit is set in the format state of the ofstream object. This constructor also sets up an associated filebuf object with a stream buffer that has length l bytes and begins at the position pointed to by p. If p is equal to 0 or l is equal to 0, the associated filebuf object is unbuffered.

AIX Considerations

This function is available for 64-bit applications. The third argument is a long value.

Overload 5

```
public:ofstream( const char* name,
          int mode = ios::out,
          int prot = filebuf::openprot,
          _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Constructs an ifstream object and opens the file name with open mode equal to mode and protection mode equal to prot, and ccsid equal to ccsid. The default value for the argument prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 6

```
public:ofstream(const char* name, int mode, _CCSID_T)
```

This is supported on **OS/400**

Constructs an ofstream object and opens the file name with open mode equal to mode and ccsid equal to ccsid. If the file cannot be opened, the error state of the constructed fstream object is set.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 7

```
public:ofstream( const char* name,
          int mode = ios::out,
          int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Constructs an ofstream object and opens the file name with open mode equal to mode and protection mode equal to prot. The default value for mode is ios::out and for prot is filebuf::openprot. If the file cannot be opened, the error state of the constructed ofstream object is set.

z/OS Considerations

The prot attribute is ignored.

Overload 8

```
public:ofstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an unopened ofstream object.

Filebuf Functions

rdbuf

```
public:filebuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the filebuf object that is attached to the ofstream object.

Open Functions

Opens the file.

z/OS Considerations

The prot attribute is ignored.

open

Opens the specified file.

Overload 1

```
public:void
  open( const char* name,
        int mode = ios::out,
        int prot = filebuf::openprot,
        _CCSID_T ccsid = _CCSID_T ( 0 ) )
```

This is supported on **OS/400**

Opens the file with the specified name, mode, protection and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file or if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 2

```
public:void
  open( const char* name,
        int mode = ios::out,
        int prot = filebuf::openprot )
```

This is supported on **WIN OS/2 AIX z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

The default value for prot is filebuf::openprot. If the fstream object is already attached to a file of if the call to fstream.rdbuf()->open() fails, ios::failbit is set in the error state for the fstream object.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

Overload 3

```
public:void open(const char* name, int mode, _CCSID_T ccsid)
```

This is supported on **OS/400**

Opens the file with the specified name, mode and coded character set id and attaches it to the fstream object.

If the file with the name, name does not already exist, open() tries to create it unless ios::nocreate is set.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

If the ccsid parameter is non-zero then it is treated as a CCSID (coded character set identifier) and will correspond to the CCSID of data written to and from the file. If the parameter value is zero then the CCSID of the job will be used.

Overload 4

```
public:void
  open( const char* name,
        const char* attr,
        int mode = ios::out,
        int prot = filebuf::openprot )
```

This is supported on **z/OS**

Opens the file with the name and attaches it to the fstream object. If the file with the name, name does not already exist, open() tries to create it with protection mode equal to prot, unless ios::nocreate is set.

You can use the attr parameter to specify additional file attributes, such as lrecl or recfm. All the parameters documented for the fopen() function are supported, with the exception of type=record.

The members of the ios::open_mode enumeration are bits that can be ORed together. The value of mode is the result of such an OR operation. This result is an int value, and for this reason, mode has type int rather than open_mode.

ofstream - Inherited Member Functions and Data

Inherited Public Functions

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~fstreambase	2927	attach	2930
close	2930	detach	2930
fstreambase	2927	open	2931
setbuf	2932		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

fstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
verify	2930		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

ostream

The ostream class lets you use the output operator << to perform formatted output, or insertion, to a stream buffer. Consider the following statement, where outs is a reference to an ostream object and x is a variable of a built-in type:

```
outs << x;
```

The output operator << calls opfx() before beginning insertion. If opfx() returns a nonzero value, the output operator converts x into a series of characters and inserts these characters into the stream buffer attached to outs. If an error occurs, the output operator sets ios::failbit.

The details of the conversion of x depend on the format state of the ostream object and the type of x. For numeric and string values, including the char* types and

wchar_t*, but excluding the char types and wchar_t, the output operator resets the width variable ios::x_width of the format state of an ostream object to 0, but it does not affect anything else in the format state.

The output operator is defined for the following types:

- Arrays of characters and char values, including arrays of wchar_t and wchar_t values
- Other integral values: short, int, long, float, double, long double, and long long values
- Pointers to void.

You can also define output operators for your own types.

Class header file: iostream.h

ostream - Hierarchy List

ios

ostream

ostream_withassign

ostream - Member Functions and Data by Group

Constructors & Destructor

Objects of the ostream class can be constructed and destructed.

~ostream

```
public:virtual ~ostream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ostream object.

ostream

Overload 1

```
public:ostream(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes a single argument which is a pointer to a streambuf object. This constructor creates an ostream object that is attached to the streambuf object pointed to by the argument. The format variables are initialized to their defaults.

Overload 2

```
public:ostream(int fd)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor is obsolete; do not use it.

Overload 3

```
public:ostream(int size, char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor is obsolete; do not use it.

Overload 4

```
protected:ostream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor is obsolete; do not use it.

Insertion Functions

You can use the insertion functions to insert characters into a stream buffer as a sequence of bytes.

complicated_put

```
public:ostream& complicated_put(char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

flush

```
public:ostream& flush()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The ultimate consumer of characters that are stored in a stream buffer may not necessarily consume them immediately. flush() causes any characters that are stored in the stream buffer attached to the output stream to be consumed.

When ostream::flush() is called, one of the following occurs:

- if the stream buffer's put area is not empty and there are characters waiting to be consumed, flush will call the stream buffer's overflow() function to flush out all the content in the put area.
- if the stream buffer's get area is not empty and there are characters waiting to be extracted, flush will call the stream buffer's sync() function. The sync() function will clean up both the put area and the get area by sending any characters that are stored in the put area to the ultimate consumer, and sending any characters that are waiting in the get area back to the ultimate producer.

ls_complicated

Overload 1

```
public:ostream& ls_complicated(char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 2

```
public:ostream& ls_complicated(signed char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

Overload 3

```
public:ostream& ls_complicated(unsigned char)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

put

```
public:ostream& put(char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts c into the stream buffer attached to the output stream. put() sets the error state of the output stream if the insertion fails.

write

Overload 1

```
public:ostream& write(const signed char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts n characters that begin at the position pointed to by s. This array of characters does not need to end with a null character.

Overload 2

```
public:ostream& write(const char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts n characters that begin at the position pointed to by s. This array of characters does not need to end with a null character.

Overload 3

```
public:ostream& write(const unsigned char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Inserts n characters that begin at the position pointed to by s. This array of characters does not need to end with a null character.

Output operators

The output operator calls the output prefix function opfx() before inserting characters into a stream buffer, and calls the output suffix function osfx() after inserting characters.

operator <<

Overload 1

```
public:ostream& operator <<(const unsigned char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts all the characters in the string into the stream buffer with the exception of the null character that terminates the string.

If ios::x_width is greater than zero and the representation of the value to be inserted is less than ios::x_width, the output operator inserts enough fill characters to ensure that the representation occupies an entire field in the stream buffer.

Overload 2

```
public:ostream& operator <<(const char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts all the characters in the string into the stream buffer with the exception of the null character that terminates the string.

If ios::x_width is greater than zero and the representation of the value to be inserted is less that ios::x_width, the output operator inserts enough fill characters to ensure that the representation occupies an entire field in the stream buffer.

Overload 3

```
public:ostream& operator <<(const void*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts pointers to void to integral values and then converts them to hexadecimal values as if ios::showbase were set. This version of the output operator is used to print out the values of pointers.

Overload 4

```
public:ostream& operator <<(ios & ( * f ) ( ios & ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The following built-in manipulators are accepted by this output operator:

```
ios&   dec(ios&)
ios&   hex(ios&)
ios&   oct(ios&)
```

These manipulators have a specific effect on an ostream object beyond inserting their own values. For example, If outs is a reference to an ostream object, then this statement sets ios::dec:

```
outs << dec;
```

Overload 5

```
public:ostream& operator <<(unsigned char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts the character into the stream buffer without performing any conversion on it.

Overload 6

```
public:ostream& operator <<(unsigned long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream

buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.

- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 7

```
public:ostream& operator <<(long long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Note: The support for long long is controlled by _LONG_LONG, __EXTENDED__, or the -q(no)longlong option.

Overload 8

```
public:ostream& operator <<(unsigned int a)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 9

```
public:ostream& operator <<(double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator performs a conversion operation on the argument and inserts it into the stream buffer attached to the output stream. The conversion depends on the values returned by the following functions:

- precision() - returns the number of significant digits that appear after the decimal. The default value is 6.
- width() - if this returns 0, the argument is inserted without any fill characters. If the return value is greater than the number of characters needed to represent the argument, extra fill characters are inserted so that the total number of characters inserted is equal to the return value.

The conversion also depends on the values of the following format flags:

- If ios::scientific is set, the argument is converted to scientific notation with one digit before the decimal, and the number of digits after the decimal equal to the value returned by precision(). The exponent begins with a lowercase ″e″ unless ios::uppercase is set, in which case the exponent begins with an uppercase ″E″.

- If ios::fixed is set, the argument is converted to fixed notation, with the number of digits after the decimal point equal to the value returned by precision().
- If neither ios::fixed nor ios::scientific is set, the conversion depends upon the value of the argument. If ios::uppercase is set, the exponents of values in scientific notation begin with an uppercase ″E″.

Overload 10

```
public:ostream& operator <<(short i)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 11

```
public:ostream& operator <<(long double)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator performs a conversion operation on the argument and inserts it into the stream buffer attached to the output stream. The conversion depends on the values returned by the following functions:

- precision() - returns the number of significant digits that appear after the decimal. The default value is 6.
- width() - if this returns 0, the argument is inserted without any fill characters. If the return value is greater than the number of

characters needed to represent the argument, extra fill characters are inserted so that the total number of characters inserted is equal to the return value.

The conversion also depends on the values of the following format flags:

- If ios::scientific is set, the argument is converted to scientific notation with one digit before the decimal, and the number of digits after the decimal equal to the value returned by precision(). The exponent begins with a lowercase ″e″ unless ios::uppercase is set, in which case the exponent begins with an uppercase ″E″.
- If ios::fixed is set, the argument is converted to fixed notation, with the number of digits after the decimal point equal to the value returned by precision().
- If neither ios::fixed nor ios::scientific is set, the conversion depends upon the value of the argument. If ios::uppercase is set, the exponents of values in scientific notation begin with an uppercase ″E″.

Overload 12

```
public:ostream& operator <<(int a)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 13

```
public:ostream& operator <<(long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 14

```
public:ostream& operator <<(unsigned long long)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Note: The support for long long is controlled by _LONG_LONG, __EXTENDED__, or the -q(no)longlong option.

Overload 15

```
public:ostream& operator <<(unsigned short i)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the integral value according to the format state of the output stream and inserts characters into the stream buffer associated with the output stream. There is no overflow detection on conversion of integral types.

The conversion that takes place depends, in part, on the settings of the following format flags:

- If ios::oct is set, the integral type is converted to a series of octal digits. If ios::showbase is set, ″0″ is inserted into the stream buffer before the octal digits. If the value being inserted is equal to 0, a single ″0″ is inserted, not ″00″.
- If ios::dec is set, the integral type is converted to a series of decimal digits.
- If ios::hex is set, the integral type is converted to a series of hexadecimal digits. If ios::showbase is set, ″0x″ (or ″0X″ if ios::uppercase is set) is inserted into the stream buffer before the hexadecimal digits.

If none of these format flags is set, the integral type is converted to a series of decimal digits. Then its sign also affects the conversion:

- If the integral type is negative, a negative sign ″-″ is inserted before the decimal digits
- If the integral type is equal to 0, the single digit 0 is inserted
- If the integral type is positive and ios::showpos is set, a positive sign ″+″ is inserted before the decimal digits.

Overload 16

```
public:ostream& operator <<(const wchar_t*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator converts the wchar_t string to its equivalent multibyte character string, and then inserts it into the stream buffer with the exception of the null character that terminates the string.

If ios::x_width is greater than zero and the representation of the value to be inserted is less than ios::x_width, the output operator inserts enough fill characters to ensure that the representation occupies an entire field in the stream buffer.

Overload 17

```
public:ostream& operator <<(signed char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts the character into the stream buffer without performing any conversion on it.

Overload 18

```
public:ostream& operator <<(float)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator performs a conversion operation on the argument and inserts it into the stream buffer attached to the output stream. The conversion depends on the values returned by the following functions:

- precision() - returns the number of significant digits that appear after the decimal. The default value is 6.
- width() - if this returns 0, the argument is inserted without any fill characters. If the return value is greater than the number of characters needed to represent the argument, extra fill characters are inserted so that the total number of characters inserted is equal to the return value.

The conversion also depends on the values of the following format flags:

- If ios::scientific is set, the argument is converted to scientific notation with one digit before the decimal, and the number of digits after the decimal equal to the value returned by precision(). The exponent begins with a lowercase ″e″ unless ios::uppercase is set, in which case the exponent begins with an uppercase ″E″.
- If ios::fixed is set, the argument is converted to fixed notation, with the number of digits after the decimal point equal to the value returned by precision().
- If neither ios::fixed nor ios::scientific is set, the conversion depends upon the value of the argument. If ios::uppercase is set, the exponents of values in scientific notation begin with an uppercase ″E″.

Overload 19

```
public:ostream& operator <<(ostream & ( * f ) ( ostream & ))
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The following built-in manipulators are accepted by this output operator:

```
ostream&   endl(ostream&)
ostream&   ends(ostream&)
ostream&   flush(ostream&)
```

These manipulators have a specific effect on an ostream object beyond inserting their own values. For example, If outs is a reference to an ostream object, then this statement inserts a newline character and calls flush():

```
outs << endl;
```

This statement inserts a null character:

```
outs << ends;
```

This statement flushes the stream buffer attached to outs. It is equivalent to flush():

```
outs << flush;
```

Overload 20

```
public:ostream& operator <<(wchar_t)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts the character into the stream buffer without performing any conversion on it.

Overload 21

```
public:ostream& operator <<(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If opfx() returns a nonzero value, the output operator inserts all of the characters that can be taken from the streambuf pointer into the stream buffer attached to the output stream. Insertion stops when no more characters can be fetched from the streambuf. No padding is performed.

Overload 22

```
public:ostream& operator <<(const signed char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts all the characters in the string into the stream buffer with the exception of the null character that terminates the string.

If ios::x_width is greater than zero and the representation of the value to be inserted is less than ios::x_width, the output operator inserts enough fill characters to ensure that the representation occupies an entire field in the stream buffer.

Overload 23

```
public:ostream& operator <<(char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The output operator inserts the character into the stream buffer without performing any conversion on it.

Positioning Functions

seekp

Functions that work with the put pointer of the ultimate consumer.

Overload 1

```
public:ostream& seekp(streampos p)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Repositions the put pointer of the ultimate consumer. Sets the put pointer to the position p.

Overload 2

```
public:ostream& seekp(streamoff o, ios::seek_dir d)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Repositions the put pointer of the ultimate consumer. Sets the put pointer to the position specified by d with the offset of o. The seek dir, d, can have the following values:

- ios::beg - the beginning of the stream
- ios::cur - the current position of the put pointer
- ios::end - the end of the stream

The new position of the put pointer is equal to the position specified by d offset by the value o. If you attempt to move the put pointer to a position that is not valid, seekp() sets ios::badbit.

tellp

```
public:streampos tellp()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the current position of the put pointer of the stream buffer that is attached to the output stream.

Prefix and Suffix Functions

Functions that are called either before or after inserting characters into the ultimate consumer.

opfx

```
public:int opfx()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

opfx() is called by the output operator before inserting characters into a stream buffer. opfx() checks the error state of the output stream. If the internal flag ios::hardfail is set, opfx() returns 0. Otherwise, opfx() flushes the stream buffer attached to the ios object pointed to by tie(), if one exists, and returns the value returned by ios::good(). ios::good() returns 0 if ios::failbit, ios::badbit, or ios:eofbit is set. Otherwise, ios::good() returns a nonzero value.

osfx

```
public:void osfx()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

osfx() is called before a formatted output function returns. osfx() flushes the streambuf object attached to the output stream if ios::unitbuf is set.

osfx() is called by the output operator. If you overload the output operator to handle your own classes, you should ensure that osfx() is called after any direct manipulation of a streambuf object. Binary output functions do not call osfx().

do_opfx

```
protected:int do_opfx()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

do_osfx

```
protected:void do_osfx()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Internal function. Do not use.

ostream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
`( * stdioflush ) ( )`	2950	`assign_private`	2941
`bp`	2941	`delbuf`	2941
`isfx_special`	2942	`ispecial`	2942
`osfx_special`	2942	`ospecial`	2942
`state`	2942	`x_fill`	2947
`x_flags`	2942	`x_precision`	2947
`x_tie`	2942	`x_width`	2947

ostream_withassign

Use this class to assign another stream to an ostream object.

Class header file: iostream.h

ostream_withassign - Hierarchy List

ios

ostream

ostream_withassign

ostream_withassign - Member Functions and Data by Group

Constructors & Destructor

Objects of the ostream_withassign class can be constructed and destructed. They can also be copied.

~ostream_withassign

```
public:virtual ~ostream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs an ostream_withassign object.

operator =

```
public:ostream_withassign& operator =(ostream_withassign& rhs)
```

This is supported on **WIN OS/2 AIX**

Copy constructor.

ostream_withassign

```
public:ostream_withassign()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an ostream_withassign object. It does not do any initialization on the object.

Assignment Operator

Assignment operators for ostream_withassign.

operator =

Overload 1

```
public:ostream_withassign& operator =(streambuf*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes a pointer to a streambuf object as its argument. It associates the streambuf with the ostream_withassign object that is on the left side of the assignment operator.

Overload 2

```
public:ostream_withassign& operator =(ostream&)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This assignment operator takes a reference to an ostream object as its argument. It associates the streambuf attached to the output stream with the ostream_withassign object that is on the left side of the assignment operator.

ostream_withassign - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

ostream			
Definition	**Page Number**	**Definition**	**Page Number**
~ostream	3001	complicated_put	3002
flush	3002	ls_complicated	3002

ostream			
Definition	**Page Number**	**Definition**	**Page Number**
operator <<	3003	opfx	3013
osfx	3013	ostream	3001
put	3003	seekp	3012
tellp	3013	write	3003

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ostream			
Definition	**Page Number**	**Definition**	**Page Number**
do_opfx	3014	do_osfx	3014
ostream	3001		

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

ostrstream

ostrstream is the class that specializes ostream to use a strstreambuf for insertion into arrays of characters in memory. You can create an ostrstream object by associating the object with a previously allocated array of characters. You can then write to it and apply other operations to it just as you would to another type of stream.

Class header file: strstream.h

ostrstream - Hierarchy List

ios

strstreambase

ostrstream

ostrstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the ostrstream class can be constructed and destructed.

~ostrstream

```
public:~ostrstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The ostrstream destructor frees space allocated by the ostrstream constructor. The destructor also writes a null byte to the stream buffer to terminate the stream.

ostrstream

Overload 1

```
public:ostrstream(signed char* str, int size, int = ios::out)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 2

```
public:ostrstream(unsigned char* str, long size, int = ios::out)
```

This is supported on **AIX**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at

the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 3

```
public:ostrstream(char* str, long size, int = ios::out)
```

This is supported on **AIX**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 4

```
public:ostrstream(signed char* str, long size, int = ios::out)
```

This is supported on **AIX**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 5

```
public:ostrstream(unsigned char* str, int size, int = ios::out)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 6

```
public:ostrstream(char* str, int size, int = ios::out)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that the stream buffer that is attached to the ostrstream object consists of an array that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the ostream::seekp() function to reposition the put pointer.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 7

```
public:ostrstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that space is allocated dynamically for the stream buffer that is attached to the ostrstream object.

Stream Buffer Functions

Use these functions to work with the stream buffer.

pcount

Returns the number of bytes that have been stored in the stream buffer. pcount() is mainly useful when binary data has been stored and the stream buffer attached to the ostrstream object is not a null-terminated string. pcount() returns the total number of bytes, not just the number of bytes up to the first null character.

Overload 1

```
public:int pcount()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

AIX Considerations

This function returns an int value for 32-bit applications. It is not available for 64-bit applications.

Overload 2

```
public:long pcount()
```

This is supported on **AIX**

AIX Considerations

This function returns a long value for 64-bit applications. It is not available for 32-bit applications.

str

```
public:char* str()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the stream buffer attached to the ostrstream and calls freeze() with a nonzero value to prevent the stream buffer from being

deleted. If the stream buffer was constructed with an explicit array, the value returned is a pointer to that array. If the stream buffer was constructed in dynamic mode, str points to the dynamically allocated area.

Until you call str(), deleting the dynamically allocated stream buffer is the responsibility of the ostrstream object. After str() has been called, the calling application has responsibility for the dynamically allocated stream buffer.

ostrstream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
rdbuf	3047		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~strstreambase	3046	strstreambase	3047

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

stdiobuf

This class is used to mix standard C input and output functions with C++ I/O Stream Library functions. This class is obsolete. New programs should avoid using this class.

Class header file: stdiostream.h

stdiobuf - Hierarchy List

streambuf
  stdiobuf

stdiobuf - Member Functions and Data by Group

Constructors & Destructor

Objects of the stdiobuf class can be constructed and destructed.

~stdiobuf

```
public:virtual ~stdiobuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructor for stdiobuf. Frees the spaces allocated by the stdiobuf constructor and flushes the file that this stdiobuf object is associated with.

stdiobuf

```
public:stdiobuf(FILE* f)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates an stdiobuf object that is associated with the FILE pointed to by f. Changes that are made to the stream buffer in an stdiobuf object are also made to the associated FILE pointed to by f.

Note: If ios::stdio is set in the format state of an ostream object, a call to osfx() flushes stdout and stderr.

Positioning Functions

overflow

```
public:virtual int overflow(int = EOF)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Emptys an output buffer. Returns EOF on error, 0 otherwise.

pbackfail

```
public:virtual int pbackfail(int c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to put back a character.

seekoff

```
public:virtual streampos seekoff(streamoff, ios::seek_dir, int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

sync

```
public:virtual int sync()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

underflow

```
public:virtual int underflow()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Fills an input buffer. Returns EOF on error or end of input, 0 otherwise.

Query Functions

stdiofile

```
public:FILE* stdiofile()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the FILE object that the stdiobuf object is associated with.

stdiobuf - Inherited Member Functions and Data

Inherited Public Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
~streambuf	3027	dbp	3031
in_avail	3028	optim_in_avail	3029
optim_sbumpc	3029	out_waiting	3036
overflow	3036	pptr_non_null	3031
sbumpc	3029	seekoff	3032
seekpos	3032	setbuf	3038
sgetc	3029	sgetn	3029
snextc	3030	sputbackc	3037
sputc	3037	sputn	3037
stossc	3033	streambuf	3028
streambuf_resource	3038	xsgetn	3030
xsputn	3038		

Inherited Public Data

None

Inherited Protected Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
allocate	3040	base	3033
blen	3041	doallocate	3041
eback	3033	ebuf	3033
egptr	3033	epptr	3033
gbump	3033	gptr	3034
pbase	3034	pbump	3034
pptr	3035	setb	3035
setg	3035	setp	3035
unbuffered	3041		

Inherited Protected Data

None

stdiostream

This class uses stdiobuf objects as stream buffers.

Class header file: stdiostream.h

stdiostream - Hierarchy List

ios

stdiostream

stdiostream - Member Functions and Data by Group

Constructors & Destructor

Objects of the stdiostream class can be constructed and destructed.

~stdiostream

```
public:~stdiostream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs a stdiostream object.

stdiostream

```
public:stdiostream(FILE*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Creates a stdiostream object that is attached to the FILE pointed to by the argument.

Miscellaneous

rdbuf

```
public:stdiobuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the stdiobuf object that is attached to the stdiostream object.

stdiostream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

streambuf

You can use the streambuf class to manipulate objects of its derived classes filebuf, stdiobuf, and strstreambuf, or to derive other classes from it.

streambuf has both a public interface and a protected interface. You should think of these two interfaces as being two separate classes, because the interfaces are used for different purposes. You should also treat streambuf as if it were defined as a virtual base class. Do not create objects of the streambuf class itself.

Although most virtual functions are declared public, you should overload them in the classes that you derive from streambuf, and consider them part of the protected interface.

Public interface

You should not create objects of the streambuf public interface directly. Instead, you should use streambuf through one of the predefined classes derived from streambuf. You can use objects of filebuf, strstreambuf and stdiobuf directly as implementations of stream buffers. The public interface consists of the streambuf public member functions that can be called on objects of these predefined classes.

streambuf itself does not have any facilities for taking characters from the ultimate producer or sending them to the ultimate consumer. The specialized member functions that handle the interface with the ultimate producer and the ultimate consumer are defined in filebuf, strstreambuf and stdiobuf.

Except for the destructor of the streambuf class, the virtual functions are described as part of the protected interface.

Protected interface

Use the streambuf protected interface in the following ways:

- As a base class to implement your own specialized stream buffers. In this sense you can think of streambuf as a virtual base class. The streambuf class only provides the basic functions needed to manipulate characters in a stream buffer. The filebuf, strstreambuf and stdiobuf classes contain functions that handle the interface with the standard ultimate consumers and producers. If you want to perform more sophisticated operations, or if you want to use other ultimate consumers and producers, you will have to create your own class derived from streambuf. You need to know about the protected interface if you want to create a class derived from streambuf.
- Through a predefined class derived from streambuf.

There are two kinds of functions in the protected interface:

- Nonvirtual member functions, which manipulate streambuf objects at a level of detail that would be inappropriate in the public interface.
- Virtual member functions, which permit classes that you derive from streambuf to customize their operations depending on the ultimate producer or ultimate consumer. When you define the virtual functions in your derived classes, you must ensure that these definitions fulfill the conditions stated in the descriptions of the virtual functions. If your definitions of the virtual functions do not fulfill these conditions, objects of the derived class may have unspecified behavior. Although most virtual functions are declared as public members, they are described with the protected interface (with the exception of the destructor for the streambuf class) because they are meant to be overridden in the classes that you derive from streambuf.

Class header file: iostream.h

streambuf - Hierarchy List

streambuf

stdiobuf

filebuf

strstreambuf

streambuf - Member Functions and Data by Group

Constructors & Destructor

Objects of the streambuf class can be constructed and destructed.

~streambuf

```
public:virtual ~streambuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The destructor for streambuf calls sync(). If a stream buffer has been set up and ios::alloc is set, sync() deletes the stream buffer.

streambuf

Overload 1

```
public:streambuf(char* p, long l)
```

This is supported on **AIX**

Constructs an empty stream buffer of length l starting at the position pointed to by p.

AIX Considerations

This constructor is available for 64-bit applications. The second argument is a long value.

Overload 2

```
public:streambuf(char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an empty stream buffer of length l starting at the position pointed to by p.

AIX Considerations

This constructor is available for 32-bit applications. The second argument is an int value.

Overload 3

```
public:streambuf(char* p, int l, int c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor is obsolete. It is included for compatibility with the AT&T C++ Language System Release 1.2. Use strstreambuf.

Overload 4

```
public:streambuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs an empty stream buffer corresponding to an empty sequence. The values returned by base(), eback(), ebuf(), egptr(), epptr(), pptr(), gptr(), and pbase() are initially all zero for this stream buffer.

Extraction Functions

Functions that extract characters from the ultimate producer, determine if characters are waiting to be extracted and handle underflow situations.

in_avail

Returns the number of characters that are available to be extracted from the get area of the stream buffer object. You can extract the number of characters equal to the value that in_avail() returns without causing an error.

Overload 1

```
public:int in_avail()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

This function returns an int value for 32-bit applications. It is not available for 64-bit applications.

Overload 2

```
public:long in_avail()
```

This is supported on **AIX**
AIX Considerations

This function returns a long value for 64-bit applications. It is not available for 32-bit applications.

optim_in_avail

```
public:int optim_in_avail()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the current get pointer is less then the end of the get area.

optim_sbumpc

```
public:int optim_sbumpc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the get pointer past one character and returns the character that it moved past.

sbumpc

```
public:int sbumpc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the get pointer past one character and returns the character that it moved past. sbumpc() returns EOF if the get pointer is already at the end of the get area.

sgetc

```
public:int sgetc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns the character after the get pointer without moving the get pointer itself. If no character is available, sgetc() returns EOF.

Note: sgetc() does not change the position of the get pointer.

sgetn

Overload 1

```
public:long sgetn(char* s, long n)
```

This is supported on **AIX**

Extracts the n characters following the get pointer, and copies them to the area starting at the position pointed to by s. If there are fewer than n characters following the get pointer, sgetn() takes the characters that are available and stores them in the position

pointed to by s. sgetn() repositions the get pointer following the extracted characters and returns the number of extracted characters.

AIX Considerations

This function is available for 64-bit applications. It accepts a long argument.

Overload 2

```
public:int sgetn(char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Extracts the n characters following the get pointer, and copies them to the area starting at the position pointed to by s. If there are fewer than n characters following the get pointer, sgetn() takes the characters that are available and stores them in the position pointed to by s. sgetn() repositions the get pointer following the extracted characters and returns the number of extracted characters.

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

snextc

```
public:int snextc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the get pointer forward one character and returns the character following the new position of the get pointer. snextc() returns EOF if the get pointer is at the end of the get area either before or after it is moved forward.

underflow

```
public:virtual int underflow()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Takes characters from the ultimate producer and puts them in the get area.

The default definition of underflow() is compatible with the AT&T C++ Language System Release 1.2 version of the stream package, but it is not considered part of the current I/O Stream Library. Thus the default definition of underflow() should not be used, and every class derived from streambuf should define underflow() itself.

If you derive underflow() in a class derived from streambuf, it should return the first character in the get area if the get area is not empty. If the get area is empty, underflow() should create a get area that is not empty and return the next character. If no more characters are available in the ultimate producer, underflow() should return EOF and leave the get area empty.

xsgetn

Overload 1

```
public:virtual int xsgetn(char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Similar to sputn.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 2

```
public:virtual long xsgetn(char* s, long n)
```

This is supported on **AIX**

Similar to sgetn.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Get/Put Pointer Functions

dbp

```
public:void dbp()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Writes to standard output the values returned by the following functions:

- base()
- eback()
- ebuf()
- egptr()
- epptr()
- gptr()
- pptr()

dbp() is intended for debugging. streambuf does not specify anything about the form of the output. dbp() is considered part of the protected interface because the information that it prints can only be understood in relation to that interface. It is declared as a public function so that it can be called anywhere during debugging.

The following example shows the output produced by dbp() when it is called as part of a filebuf object:

```
#include < iostream.h >
int main()
{
cout << "Here is some sample output." << endl;
cout.rdbuf()->dbp();
}
```

If you compile and run this example, your output will look like this:

```
Here is some sample output.
buf at 0x90210, base=0x91010, ebuf=0x91410,
pptr=0x91010, epptr=0x91410, eback=0, gptr=0, egptr=0
```

pptr_non_null

```
public:int pptr_non_null()
```

This is supported on **AIX**

Returns true if the put pointer is not null.

seekoff

```
public:virtual streampos
  seekoff( streamoff,
           ios::seek_dir,
           int = ios::in|ios::out )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Repositions the get or put pointer of the ultimate producer or ultimate consumer. seekoff() does not change the values returned by gptr() or pptr().

The default definition of seekoff() returns EOF.

If you define your own seekoff() function, it should return EOF if the derived class does not support repositioning. If the class does support repositioning, seekoff() should return the new position of the affected pointer, or EOF if an error occurs.

The first argument is an offset from a position in the ultimate producer or ultimate consumer. The second argument is a position in the ultimate produce or ultimate consumer. It can have the following values:

- ios::beg - the beginning of the ultimate producer or consumer
- ios::cur - the current position in the ultimate producer or consumer
- ios::end - the end of the ultimate producer or consumer

The new position of the affected pointer is the position specified by the seek dir offset by the value of the stream offset. If you derive your own classes from streambuf, certain values of the seek dir may not be valid depending on the nature of the ultimate consumer or producer.

If ios::in is set in the third argument, the seekoff() should modify the get pointer. If ios::out is set, the put pointer should be modified. If both ios::in and ios::out are set, both the get pointer and the put pointer should be modified.

seekpos

```
public:virtual streampos
  seekpos( streampos,
           int = ios::in|ios::out )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Repositions the get or put pointer of the ultimate producer or consumer to the streampos position. If ios::in is set, the get pointer is repositioned. If ios::out is set, the put pointer is repositioned. If both ios::in and ios::out are set, both the get pointer and the put pointer are affected. seekpos() does not change the values returned by gptr() or pptr().

The default definition of seekpos() returns the return value of the function seekoff(streamoff(pos), ios::beg, mode). Thus, if you want to define seeking operations in a class derived from streambuf, you can define seekoff() and use the default definition of seekpos().

If you define seekpos() in a class derived from streambuf, seekpos() should return EOF if the class does not support repositioning or if the streampos points to a position equal to or greater than the end of the stream. If not, seekpos() should return the streampos.

stossc

```
public:void stossc()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the get pointer forward one character. If the get pointer is already at the end of the get area, stossc() does not move it.

base

```
protected:char* base()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the first byte of the stream buffer. The stream buffer consists of the space between the pointer returned by base() and the pointer returned by ebuf().

eback

```
protected:char* eback()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the lower bound of the space available for the get area of the streambuf. The space between the pointer returned by eback() and the pointer returned by gptr() is available for putback.

ebuf

```
protected:char* ebuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the byte after the last byte of the stream buffer.

egptr

```
protected:char* egptr()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the byte after the last byte of the get area of the streambuf.

epptr

```
protected:char* epptr()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the byte after the last byte of the put area of the streambuf.

gbump

Overload 1

```
protected:void gbump(long n)
```

This is supported on **AIX**

Offsets the beginning of the get area by the value of n. The value of n can be positive or negative. gbump() does not check to see if the new value returned by gptr() is valid.

The beginning of the get area is equal to the value returned by gptr().

AIX Considerations

This function is available for 64-bit applications. It accepts a long argument.

Overload 2

```
protected:void gbump(int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Offsets the beginning of the get area by the value of n. The value of n can be positive or negative. gbump() does not check to see if the new value returned by gptr() is valid.

The beginning of the get area is equal to the value returned by gptr().

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

gptr

```
protected:char* gptr()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the first byte of the get area of the streambuf. The get area consists of the space between the pointer returned by gptr() and the pointer returned by egptr(). Characters are extracted from the stream buffer beginning at the character pointed to be gptr().

pbase

```
protected:char* pbase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the beginning of the space available for the put area of the streambuf. Characters between the pointer returned by pbase() and the pointer returned by pptr() have been stored in the stream buffer, but they have not been consumed by the ultimate consumer.

pbump

Overload 1

```
protected:void pbump(long n)
```

This is supported on **AIX**

Offsets the beginning of the put area by the value of n. The value of n can be positive or negative. pbump() does not check to see if the new value returned by pptr() is valid.

The beginning of the put area is equal to the value returned by pptr().

AIX Considerations

This function is available for 64-bit applications. It accepts a long argument.

Overload 2

```
protected:void pbump(int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Offsets the beginning of the put area by the value of n. The value of n can be positive or negative. pbump() does not check to see if the new value returned by pptr() is valid.

The beginning of the put area is equal to the value returned by pptr().

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

pptr

```
protected:char* pptr()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the beginning of the put area of the streambuf. The put area consists of the space between the pointer returned by pptr() and the pointer returned by epptr().

setb

```
protected:void setb(char* b, char* eb, int a = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the beginning of the existing stream buffer (the pointer returned by base()) to the position pointed to by b, and sets the end of the stream buffer (the pointer returned by ebuf()) to the position pointed to by eb.

If a is a nonzero value, the stream buffer will be deleted when setb() is called again. If b and eb are both equal to 0, no stream buffer is established. If b is not equal to 0, a stream buffer is established, even if eb is less than b. If this is the case, the stream buffer has length zero.

setg

```
protected:void setg(char* eb, char* g, char* eg)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the beginning of the get area of streambuf (the pointer returned by gptr()) to g, and sets the end of the get area (the pointer returned by egptr()) to eg. setg() also sets the beginning of the area available for putback (the pointer returned by eback()) to eb.

setp

```
protected:void setp(char* p, char* ep)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets the spaces available for the put area. Both the start (pbase()) and the beginning (pptr()) of the put area are set to the value p.

Sets the beginning of the put area of the streambuf (the pointer returned by pptr()) to the position pointed to by p, and sets the end of the put area (the pointer returned by epptr()) to the position pointed to by ep.

Insertion Functions

Functions that insert characters into the ultimate consumer, determine if characters are waiting to be inserted and handle overflow situations.

out_waiting

Returns the number of characters that are in the put area waiting to be sent to the ultimate consumer.

Overload 1

```
public:int out_waiting()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

This function returns an int value for 32-bit applications. It is not available for 64-bit applications.

Overload 2

```
public:long out_waiting()
```

This is supported on **AIX**
AIX Considerations

This function returns a long value for 64-bit applications. It is not available for 32-bit applications.

overflow

```
public:virtual int overflow(int c = EOF)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called when the put area is full, and an attempt is made to store another character in it. overflow() may be called at other times.

The default definition of overflow() is compatible with the AT&T C++ Language System Release 1.2 version of the stream package, but it is not considered part of the current I/O Stream Library. Thus, the default definition of overflow() should not be used, and every class derived from streambuf should define overflow() itself.

The definition of overflow() in your classes derived from streambuf should cause the ultimate consumer to consume the characters in the put area, call setp() to establish a new put area, and store the argument c in the put area if c does not equal EOF. overflow() should return EOF if an error occurs, and it should return a value not equal to EOF otherwise.

pbackfail

```
public:virtual int pbackfail(int c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called when both of the following conditions are true:

- An attempt has been made to put back a character.
- There is no room in the putback area. The pointer returned by eback() equals the pointer returned by gptr().

The default definition of pbackfail() returns EOF.

If you define pbackfail() in your own classes, your definition of pbackfail() should attempt to deal with the full putback area by, for instance, repositioning the get pointer of the ultimate producer. If this is possible, pbackfail() should return the argument c. If not, pbackfail() should return EOF.

sputbackc

```
public:int sputbackc(char c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Moves the get pointer back one character. The get pointer may simply move, or the ultimate producer may rearrange the internal data structures so that the character c is saved. The argument c must equal the character that precedes the get pointer in the stream buffer. The effect of sputbackc() is undefined if c is not equal to the character before the get pointer. sputbackc() returns EOF if an error occurs. The conditions that cause errors depend on the derived class.

sputc

```
public:int sputc(int c)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stores the argument c after the put pointer and moves the put pointer past the stored character. If there is enough space in the stream buffer, this will extend the size of the put area. sputc() returns EOF if an error occurs. The conditions that cause errors depend on the derived class.

sputn

Overload 1

```
public:int sputn(const char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Stores the n characters starting at s after the put pointer and moves the put pointer to the end of the series. sputn() returns the number of characters successfully stored. If an error occurs, sputn() returns a value less than n.

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

Overload 2

```
public:long sputn(const char* s, long n)
```

This is supported on **AIX**

Stores the n characters starting at s after the put pointer and moves the put pointer to the end of the series. sputn() returns the number of characters successfully stored. If an error occurs, sputn() returns a value less than n.

AIX Considerations

This function is available for 64-bit applications. It accepts a long argument.

xsputn

Overload 1

```
public:virtual int xsputn(const char* s, int n)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Similar to sputn.
AIX Considerations

This function is available for use when building 32-bit applications. The second argument is an int value.

Overload 2

```
public:virtual long xsputn(const char* s, long n)
```

This is supported on **AIX**

Similar to sputn.
AIX Considerations

This function is available for use when building 64-bit applications. The second argument is a long value.

Locking functions

streambuf_resource

```
public:IRTLResource& streambuf_resource() const
```

This is supported on **z/OS**

Stream Buffer Functions

Functions that work with the underlying streambuf object.
setbuf

Overload 1

```
public:streambuf* setbuf(unsigned char* p, long len)
```

This is supported on **AIX**

Sets up a stream buffer consisting of the array of bytes starting at p with length len.

This function is different from setb(). setb() sets pointers to an existing stream buffer. setbuf(), however, creates the stream buffer.

The default definition of setbuf() sets up the stream buffer if the streambuf object does not already have a stream buffer.

If you define setbuf() in a class derived from streambuf, setbuf() can either accept or ignore a request for an unbuffered streambuf object. The call to setbuf() is a request for an unbuffered streambuf object if p equals 0 or len equals 0. setbuf() should return a pointer to the streambuf if it accepts the request, and 0 otherwise.
AIX Considerations

This function is available for 64-bit applications. It accepts an long argument.

Overload 2

```
public:virtual streambuf* setbuf(char* p, long len)
```

This is supported on **AIX**

Sets up a stream buffer consisting of the array of bytes starting at p with length len.

This function is different from setb(). setb() sets pointers to an existing stream buffer. setbuf(), however, creates the stream buffer.

The default definition of setbuf() sets up the stream buffer if the streambuf object does not already have a stream buffer.

If you define setbuf() in a class derived from streambuf, setbuf() can either accept or ignore a request for an unbuffered streambuf object. The call to setbuf() is a request for an unbuffered streambuf object if p equals 0 or len equals 0. setbuf() should return a pointer to the streambuf if it accepts the request, and 0 otherwise.

AIX Considerations

This function is available for 64-bit applications. It accepts an long argument.

Overload 3

```
public:streambuf* setbuf(char* p, int len, int count)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This function is obsolete. The I/O Stream Library includes it to be compatible with AT&T C++ Language System Release 1.2

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

Overload 4

```
public:virtual streambuf* setbuf(char* p, int len)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets up a stream buffer consisting of the array of bytes starting at p with length len.

This function is different from setb(). setb() sets pointers to an existing stream buffer. setbuf(), however, creates the stream buffer.

The default definition of setbuf() sets up the stream buffer if the streambuf object does not already have a stream buffer.

If you define setbuf() in a class derived from streambuf, setbuf() can either accept or ignore a request for an unbuffered streambuf object. The call to setbuf() is a request for an unbuffered streambuf object if p equals 0 or len equals 0. setbuf() should return a pointer to the streambuf if it accepts the request, and 0 otherwise.

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

Overload 5

```
public:streambuf* setbuf(unsigned char* p, int len)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Sets up a stream buffer consisting of the array of bytes starting at p with length len.

This function is different from setb(). setb() sets pointers to an existing stream buffer. setbuf(), however, creates the stream buffer.

The default definition of setbuf() sets up the stream buffer if the streambuf object does not already have a stream buffer.

If you define setbuf() in a class derived from streambuf, setbuf() can either accept or ignore a request for an unbuffered streambuf object. The call to setbuf() is a request for an unbuffered streambuf object if p equals 0 or len equals 0. setbuf() should return a pointer to the streambuf if it accepts the request, and 0 otherwise.

AIX Considerations

This function is available for 32-bit applications. It accepts an int argument.

sync

```
public:virtual int sync()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Synchronizes the stream buffer with the ultimate producer or the ultimate consumer.

The default definition of sync() returns 0 if either of the following conditions is true:

- The get area is empty and there are no characters waiting to go to the ultimate consumer.
- No stream buffer has been allocated for the streambuf.

Otherwise, sync() returns EOF.

If you define sync() in a class derived from streambuf, it should send any characters that are stored in the put area to the ultimate consumer, and (if possible) send any characters that are waiting in the get area back to the ultimate producer. When sync() returns, both the put area and the get area should be empty. sync() should return EOF if an error occurs.

allocate

```
protected:int allocate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to set up a stream buffer. allocate() returns the following values:

- 0, if the streambuf has a stream buffer set up (that is, base() returns a nonzero value), or if unbuffered() returns a nonzero value. allocate() does not do any further processing if it returns 0.
- 1, if allocate() does set up a stream buffer.
- EOF, if the attempt to allocate space for the stream buffer fails.

allocate() is not called by any other nonvirtual member function of streambuf.

blen

Returns the length (in bytes) of the stream buffer.

Overload 1

```
protected:long blen() const
```

This is supported on **AIX**
AIX Considerations

The value returned is a long when building 64-bit aplications. This function is not available for 32-bit applications.

Overload 2

```
protected:int blen() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
AIX Considerations

The value returned is an int when building 32-bit applications. This function is not available for 64-bit applications.

doallocate

```
protected:virtual int doallocate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Called when allocate() determines that space is needed for a stream buffer.

The default definition of doallocate() attempts to allocate space for a stream buffer using the operator new.

If you define your own version of doallocate(), it must call setb() to provide space for a stream buffer or return EOF if it cannot allocate space. doallocate() should only be called if unbuffered() and base() return zero.

In your own version of doallocate(), you provide the size of the buffer for your constructor. Assign the buffer size you want to a variable using a #define statement. This variable can then be used in the constructor for your doallocate() function to define the size of the buffer.

unbuffered

Overload 1

```
protected:void unbuffered(int unb)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Manipulates the private streambuf variable called the buffering state. If the buffering state is nonzero, a call to allocate() does not set up a stream buffer.

Changes the value of the buffering state to unb.

Overload 2

```
protected:int unbuffered() const
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Manipulates the private streambuf variable called the buffering state. If the buffering state is nonzero, a call to allocate() does not set up a stream buffer.

Returns the current value of the buffering state.

streambuf - Inherited Member Functions and Data

Inherited Public Functions

None

Inherited Public Data

None

Inherited Protected Functions

None

Inherited Protected Data

None

---

strstream

strstream is the class that specializes iostream to use a strstreambuf for input and output with arrays of characters in memory. You can create an strstream object by associating the object with a previously allocated array of characters. You can then write output to it, read input from it, and apply other operations to it just as you would to another type of stream.

Class header file: strstream.h

strstream - Hierarchy List

ios

strstreambase

strstream

strstream - Member Functions and Data by Group

Constructors & Destructor

Objects of the strstream class can be constructed and destructed.

~strstream

```
public:~strstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

The strstream destructor frees the space allocated by the strstream constructor.

strstream

Overload 1

```
public:strstream(char* str, long size, int mode)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to

by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 2

```
public:strstream(char* str, int size, int mode)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 3

```
public:strstream(signed char* str, long size, int mode)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 4

```
public:strstream(unsigned char* str, int size, int mode)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 5

```
public:strstream(signed char* str, int size, int mode)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 6

```
public:strstream(unsigned char* str, long size, int mode)
```

This is supported on **AIX**

This constructor specifies that characters should be extracted and inserted into the array of bytes that starts at the position pointed to by str with a length of size bytes. If ios::ate or ios::app is set in mode, str points to a null-terminated string and insertions begin at the null character. Otherwise, insertions begin at the position pointed to by str. You can use the istream::seekg() function to reposition the get pointer anywhere in this array.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 7

```
public:strstream()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes no arguments and specifies that space is allocated dynamically for the stream buffer that is attached to the strstream object.

Stream Buffer Functions

str

```
public:char* str()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the stream buffer attached to the strstream and calls freeze() with a nonzero value to prevent the stream buffer from being deleted. If the stream buffer was constructed with an explicit array, the value returned is a pointer to that array. If the stream buffer was constructed in dynamic mode, str points to the dynamically allocated area.

Until you call str(), deleting the dynamically allocated stream buffer is the responsibility of the strstream object. After str() has been called, the calling application has responsibility for the dynamically allocated stream buffer.

Note: If your application calls str() without calling freeze() with a nonzero argument (to unfreeze the strstream), or without explicitly deleting the array of characters returned by the call to str(), the array of characters will not be deallocated by the strstream when it is destroyed. This situation is a potential source of a memory leak.

strstream - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
rdbuf	3047		

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

strstreambase			
Definition	**Page Number**	**Definition**	**Page Number**
~strstreambase	3046	strstreambase	3047

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init setstate	2947 2943	ios	2941

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( ) bp isfx_special osfx_special state x_flags x_tie	2950 2941 2942 2942 2942 2942 2942	assign_private delbuf ispecial ospecial x_fill x_precision x_width	2941 2941 2942 2942 2947 2947 2947

strstreambase

The strstreambase class is an internal class that provides common functions for the classes that are derived from it; strstream, istrstream, and ostrstream. Do not use the strstreambase class directly.

Class header file: strstream.h

strstreambase - Hierarchy List

ios
- **strstreambase**
  - istrstream
  - ostrstream
  - strstream

strstreambase - Member Functions and Data by Group

Constructors & Destructor

Objects of the strstreambase class can be constructed and destructed by objects derived from it. Do not use these functions directly.

~strstreambase

```
protected:~strstreambase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Destructs a strstreambase object.

strstreambase

Overload 1

```
protected:strstreambase(char*, long, char*)
```

This is supported on **AIX**

Constructs a strstreambase object.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 2

```
protected:strstreambase(char*, int, char*)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a strstreambase object.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

Overload 3

```
protected:strstreambase()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a strstreambase object.

Misc

rdbuf

```
public:strstreambuf* rdbuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the stream buffer that the strstreambase object is attached to.

strstreambase - Inherited Member Functions and Data

Inherited Public Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
~ios	2940	bad	2942
bitalloc	2947	clear	2943
eof	2943	fail	2943
fill	2944	flags	2944

ios			
Definition	**Page Number**	**Definition**	**Page Number**
good	2943	ios	2941
ios_resource	2947	iword	2948
operator !	2948	operator const void *	2948
operator void *	2948	precision	2944
pword	2948	rdbuf	2948
rdstate	2943	setf	2945
skip	2946	sync_with_stdio	2949
tie	2949	unsetf	2946
width	2946	xalloc	2950

Inherited Public Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
adjustfield	2941	basefield	2941
floatfield	2941		

Inherited Protected Functions

ios			
Definition	**Page Number**	**Definition**	**Page Number**
init	2947	ios	2941
setstate	2943		

Inherited Protected Data

ios			
Definition	**Page Number**	**Definition**	**Page Number**
(* stdioflush) ( )	2950	assign_private	2941
bp	2941	delbuf	2941
isfx_special	2942	ispecial	2942
osfx_special	2942	ospecial	2942
state	2942	x_fill	2947
x_flags	2942	x_precision	2947
x_tie	2942	x_width	2947

strstreambuf

This class specializes streambuf to use an array of bytes in memory as the source or target of data.

Class header file: strstream.h

strstreambuf - Hierarchy List

streambuf

strstreambuf

strstreambuf - Member Functions and Data by Group

Constructors & Destructor

Objects of the strstreambuf class can be constructed and destructed.

~strstreambuf

```
public:~strstreambuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If freeze() has not been called for the strstreambuf object and a stream buffer is associated with the strstreambuf object, the strstreambuf destructor frees the space allocated by the strstreambuf constructor. The effect of the destructor depends on the constructor used to create the strstreambuf object:

- If you created the strstreambuf object using the constructor that takes two pointers to functions as arguments, the destructor frees the space allocated by the destructor by calling the function pointed to by the second argument to the constructor.
- If you created the strstreambuf object using any of the other constructors, the destructor calls the delete operator to free the space allocated by the constructor.

strstreambuf

Overload 1

```
public:strstreambuf(long)
```

This is supported on **AIX**

This constructor takes one argument and constructs an empty strstreambuf object in dynamic mode. The initial size of the stream buffer will be at least as long as the argument in bytes.

AIX Considerations

This constructor is available for 64-bit applications. It accepts a long argument.

Overload 2

```
public:strstreambuf(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes one argument and constructs an empty strstreambuf object in dynamic mode. The initial size of the stream buffer will be at least as long as the argument in bytes.

AIX Considerations

This constructor is available for 32-bit applications. It accepts an int argument.

Overload 3

```
public:strstreambuf(char* b, int size, char* pstart = 0)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.
- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 32-bit applications. The second argument is an int value.

Overload 4

```
public:strstreambuf( unsigned char* b,
             int size,
             unsigned char* pstart = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.
- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 32-bit applications. The second argument is an int value.

Overload 5

```
public:strstreambuf( unsigned char* b,
             long size,
             unsigned char* pstart = 0 )
```

This is supported on **AIX**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.
- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 64-bit applications. The second argument is a long value.

Overload 6

```
public:strstreambuf( void * ( * a ) ( long ),
             void ( * f ) ( void * ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes two arguments and creates an empty strstreambuf object in dynamic mode. a is a pointer to the function that is used to allocate space. a is passed a long value that equals the number of bytes that it is supposed to allocate. If the value of a is 0, the operator new is used to allocate space. f is a pointer to the function that is used to free space. f is passed an argument that is a pointer to the array of bytes that a allocated. If f has a value of 0, the operator delete is used to free space.

Overload 7

```
public:strstreambuf( signed char* b,
             int size,
             signed char* pstart = 0 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.

- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 32-bit applications. The second argument is an int value.

Overload 8

```
public:strstreambuf( signed char* b,
             long size,
             signed char* pstart = 0 )
```

This is supported on **AIX**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.
- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 64-bit applications. The second argument is a long value.

Overload 9

```
public:strstreambuf(char* b, long size, char* pstart = 0)
```

This is supported on **AIX**

Constructs a strstreambuf object with a stream buffer that begins at the position pointed to by b. The nature of the stream buffer depends on the value of size.

- If size is positive, the size bytes following the position pointed to by b make up the stream buffer.
- If size equals 0, b points to the beginning of a null-terminated string, and the bytes of that string, excluding the terminating null character, will make up the stream buffer.
- If size is negative, the stream buffer has an indefinite length. The get pointer of the stream buffer is initialized to b, and the put pointer is initialized to pstart.

Regardless of the values of size, if the value of pstart is 0, the get area will include the entire stream buffer, and insertions will caues errors.

AIX Considerations

This constructor is available for 64-bit applications. The second argument is a long value.

Overload 10

```
public:strstreambuf()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

This constructor takes no arguments and constructs an empty strstreambuf object in dynamic mode. Space will be allocated automatically to accommodate the characters that are put into the strstreambuf objet. This space will be allocated using the operator new and deallocated using the operator delete. The characters that are already stored by the strstreambuf object may have to be copied when new space is allocated. If you know you are going to insert many characters into an strstreambuf object, you can give the I/O Stream Library an estimate of the size of the object before you create it by calling strstreambuf::setbuf().

Get/Put Pointer Functions

seekoff

```
public:virtual streampos seekoff(streamoff, ios::seek_dir, int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Repositions the get or put pointer in the array of bytes in memory that serves as the ultimate producer or consumer.

If you constructed the strstreambuf in dynamic mode, the results of seekoff() are unpredictable. Therefore, do not use seekoff() with an strstreambuf object that you created in dynamic mode.

If you did not construct the strstreambuf object in dynamic mode, seekoff() attempts to reposition the get pointer or the put pointer, depending on the value of the third argument, the mode. If ios::in is set, seekoff() repositions the get pointer. If ios::out is set, seekoff() repositions the put pointer. If both ios::in and ios::out are set, seekoff() repositions both pointers.

seekoff() attempts to reposition the affected pointer to the value of ios::seek_dir + streamoff. ios::seek_dir can have the following values: ios::beg, ios::cur, or ios::end.

If the value of ios::seek_dir + streamoff is equal to or greater than the end of the array, the value is not valid and seekoff() returns EOF. Otherwise, seekoff() sets the affected pointer to this value and returns this value.

Insertion & Extraction Functions

overflow

```
public:virtual int overflow(int)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Causes the ultimate consumer to consume the characters in the put area and calls setp() to establish a new put area. The argument is stored in the new put area if its value is not equal to EOF.

pcount

This function is internal and should not be used.

Overload 1

```
public:long pcount()
```

This is supported on **AIX**

AIX Considerations

This function returns a long for 64-bit applications. It is not available for 32-bit applications.

Overload 2

```
public:int pcount()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

AIX Considerations

This function returns an int for 32-bit applications. It is not available for 64-bit applications.

underflow

```
public:virtual int underflow()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

If the get area is not empty, underflow() returns the first character in the get area. If the get area is empty, underflow() creates a new get area that is not empty and returns the first character. If no more characters are available in the ultimate producer, underflow() returns EOF and leaves the get area empty.

Stream Buffer Functions

doallocate

```
public:virtual int doallocate()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Attempts to allocate space for a stream buffer. If you created the strstreambuf object using the constructor that takes two pointers to functions as arguments, doallocate() allocates space for the stream buffer by calling the function pointed to by the first argument to the constructor. Otherwise, doallocate() calls the operator new to allocate space for the stream buffer.

freeze

```
public:void freeze(int n = 1)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Controls whether the array that makes up a stream buffer can be deleted automatically. If n has a nonzero value, the array is not deleted automatically. If n equals 0, the array is deleted automatically when more space is needed or when the strstreambuf object is deleted. If you call freeze() with a nonzero argument for a strstreambuf object that was allocated in dynamic mode, any attempts to put characters in the stream

buffer may result in errors. Therefore, you should avoid insertions to such stream buffers because the results are unpredictable. However, if you have a "frozen" stream buffer and you call freeze() with an argument equal to 0, you can put characters in the stream buffer again.

Only space that is acquired through dynamic allocation is ever freed.

isfrozen

```
public:int isfrozen()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns true if the stream buffer is frozen.

setbuf

Overload 1

```
public:virtual streambuf* setbuf(char* p, long l)
```

This is supported on **AIX**

setbuf() records the buffer size. The next time that the strstreambuf object dynamically allocates a stream buffer, the stream buffer is at least l bytes long.

Note: If you call setbuf() for an strstreambuf object, you must call it with the first argument equal to 0.

AIX Considerations

This function is available for 64-bit applications. The second argument is a long value.

Overload 2

```
public:virtual streambuf* setbuf(char* p, int l)
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

setbuf() records the buffer size. The next time that the strstreambuf object dynamically allocates a stream buffer, the stream buffer is at least l bytes long.

Note: If you call setbuf() for an strstreambuf object, you must call it with the first argument equal to 0.

AIX Considerations

This function is available for 32-bit applications. The second argument is an int value.

str

```
public:char* str()
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Returns a pointer to the first character in the stream buffer and calls freeze() with a nonzero argument. Any attempts to put characters in the stream buffer may result in errors. If the strstreambuf object was created with an explicit array (that is, the strstreambuf constructor with three arguments was used), str() returns a pointer to that array. If the strstreambuf object was created in dynamic mode and nothing is stored in the array, str() may return 0.

strstreambuf - Inherited Member Functions and Data

Inherited Public Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
~streambuf	3027	dbp	3031
in_avail	3028	optim_in_avail	3029
optim_sbumpc	3029	out_waiting	3036
overflow	3036	pbackfail	3036
pptr_non_null	3031	sbumpc	3029
seekoff	3032	seekpos	3032
setbuf	3038	sgetc	3029
sgetn	3029	snextc	3030
sputbackc	3037	sputc	3037
sputn	3037	stossc	3033
streambuf	3028	streambuf_resource	3038
sync	3040	xsgetn	3030
xsputn	3038		

Inherited Public Data

None

Inherited Protected Functions

streambuf			
Definition	**Page Number**	**Definition**	**Page Number**
allocate	3040	base	3033
blen	3041	eback	3033
ebuf	3033	egptr	3033
epptr	3033	gbump	3033
gptr	3034	pbase	3034
pbump	3034	pptr	3035
setb	3035	setg	3035
setp	3035	unbuffered	3041

Inherited Protected Data

None

Chapter 11. Global Enums, Typedefs, and Variables

Global Enumerations

BooleanConstants

```
enum BooleanConstants { false=0,
                        true=1 }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IExplicitInit

```
enum IExplicitInit { IINIT }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITreeIterationOrder

```
enum ITreeIterationOrder { IPreorder,
                           IPostorder }
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Global Type Definitions

bool

```
typedef int
        bool
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

BooleanConstants

```
typedef bool
        BooleanConstants
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

GCoord

```
typedef double
        GCoord
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IBool

```
typedef int
        IBool
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEventParameter1

```
typedef IEventData
        IEventParameter1
```

Use the IEventParameter1 class to encapsulate the message parameter (MPARAM) an result (MRESULT) data of an event.

IEventParameter1 is an alias of the IEventData class. See IEventData for more information.

In addition, IEventParameter1 is interchangable with the following classes (which are other aliases of the class IEventData):

IEventParameter2

IEventResult

IHighEventParameter

© Copyright IBM Corp. 2001

ILowEventParameter

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEventParameter2

```
typedef IEventData
        IEventParameter2
```

Use the IEventParameter2 class to encapsulate the message parameter (MPARAM) an result (MRESULT) data of an event.

IEventParameter2 is an alias of the IEventData class. See IEventData for more information.

In addition, IEventParameter2 is interchangable with the following classes (which are other aliases of the class IEventData):

IEventParameter1

IEventResult

IHighEventParameter

ILowEventParameter

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IEventResult

```
typedef IEventData
        IEventResult
```

Use the IEventResult class to encapsulate the message parameter (MPARAM) and message result (MRESULT) data of an event.

IEventResult is an alias of the IEventData class. See IEventData for more information.

In addition, IEventResult is interchangable with the following classes (which are other aliases of the class IEventData):

IEventParameter1

IEventParameter2

IHighEventParameter

ILowEventParameter

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFifoIOStream

```
typedef IFifoUslIOStream
        IFifoIOStream
```

This is supported on **AIX OS/400 z/OS**

IFifoIOStream

```
typedef IFifoStdIOStream
        IFifoIOStream
```

This is supported on **AIX OS/400 z/OS**

IFileIOStream

```
typedef IFileUslIOStream
        IFileIOStream
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFileIOStream

```
typedef IFileStdIOStream
        IFileIOStream
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFileKind

```
typedef const void *
        IFileKind
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFileName

```
typedef IText
        IFileName
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IFileSize

```
typedef ILongLong
        IFileSize
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHighEventParameter

```
typedef IEventData
        IHighEventParameter
```

Use the IHighEventParameter class to encapsulate the message parameter (MPARAM) result (MRESULT) data of an event.

IHighEventParameter is an alias of the IEventData class. See IEventData for more information.

In addition, IHighEventParameter is interchangable with the following classes (which are other aliases of the class IEventData):

IEventParameter1

IEventParameter2

IEventResult

ILowEventParameter

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IHostPathName

```
typedef IText
        IHostPathName
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ILongLong

```
typedef __int64
        ILongLong
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ILongLong

```
typedef long long
        ILongLong
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ILowEventParameter

```
typedef IEventData
        ILowEventParameter
```

Use the ILowEventParameter class to encapsulate the message parameter (MPARAM) result (MRESULT) data of an event.

ILowEventParameter is an alias of the IEventData class. See IEventData for more information.

In addition, ILowEventParameter is interchangable with the following classes (which are other aliases of the class IEventData):

IEventParameter1

IEventParameter2

IEventResult

IHighEventParameter

This is supported on **WIN OS/2 AIX OS/400 z/OS**

INotificationId

```
typedef const char *
        INotificationId
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

INumber

```
typedef ioc::unumeric_t
        INumber
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IPosition

```
typedef ioc::unumeric_t
        IPosition
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IReverseConstTextIterator

```
typedef IReverseIterator < IConstTextIterator , ioc::unichar_t >
        IReverseConstTextIterator
```

IReverseConstTextIterator is the same as IConstTextIterator, except that its ++, --, +=, -=, +, and - operators all move in the opposite of their normal direction. For example, if you have an IConstTextIterator and an IConstReverseTextIterator that refer to the same character, caling ++ on the reverse iterator will point it at the same character as calling -- on the forward iterator. This reversal of direction does not affect the comparison operators, or the - operator that takes two iterators. In other words, if you code the following:

```
IReverseConstTextIterator b(a);
b += 2;
```

″b < a″ would now return true.

The text() and offset() functions are not supported on IReverseConstTextIterator, but an IReverseConstTextIterator can be converted to an IConstTextIterator. **Note:** IReverseConstTextIterator is currently typedefed as an instantiation of the template class

IReverseIterator. Do not use IReverseIterator yourself. It is not considered API and may be replaced with something else in the future.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IReverseConstTextIterator

```
typedef IReverseIterator < IConstTextIterator , ioc::unichar_t >
        IReverseConstTextIterator
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IReverseTextIterator

```
typedef IReverseIterator < ITextIterator , ICharacterReference >
        IReverseTextIterator
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IReverseTextIterator

```
typedef IReverseIterator < ITextIterator , ICharacterReference >
        IReverseTextIterator
```

IReverseConstTextIterator is the same as IConstTextIterator, except that its ++, --, +=, -=, +, and - operators all move in the opposite of their normal direction. For example, if you have an IConstTextIterator and an IConstReverseTextIterator that refer to the same character, caling ++ on the reverse iterator will point it at the same character as calling -- on the forward iterator. This reversal of direction does not affect the comparison operators, or the - operator that takes two iterators. In other words, if you code the following:

```
IReverseConstTextIterator b(a);
b += 2;
```

″b < a″ would now return true.

The text() and offset() functions are not supported on IReverseConstTextIterator, but an IReverseConstTextIterator can be converted to an IConstTextIterator. **Note:** IReverseConstTextIterator is currently typedefed as an instantiation of the template class IReverseIterator. Do not use IReverseIterator yourself. It is not considered API and may be replaced with something else in the future.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ISemaphoreHandle

```
typedef ISharedSemaphoreHandle
        ISemaphoreHandle
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ITestDecisionFn

```
typedef bool ( ITest::* ITestDecisionFn)( )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

IThreadHandle

```
typedef IThreadId
        IThreadHandle
```

This is supported on **OS/2 AIX OS/400 z/OS**

IVCollection

```
typedef IVCollectionImpl
        IVCollection
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

SIZEL

```
typedef struct _SIZEL
        SIZEL
```

This is supported on **OS/2**

SIZEL

```
typedef struct tagSIZE
        SIZEL
```

This is supported on **WIN**

streamoff

```
typedef long
        streamoff
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

streampos

```
typedef long
        streampos
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Global Variables

gInvalidThreadSlotIndex

```
const ioc::unumeric_t gInvalidThreadSlotIndex = 0xFFFFFFFFFFFFFFFFUL
```

This is supported on **WIN OS/2 AIX z/OS**

gInvalidThreadSlotIndex

```
const ioc::unumeric_t gInvalidThreadSlotIndex = ULONGLONG_MAX
```

This is supported on **OS/400**

gInvalidThreadSlotIndex

```
const ioc::unumeric_t gInvalidThreadSlotIndex = 0xFFFFFFFFUL
```

This is supported on **WIN OS/2 AIX z/OS**

Iostream_init

```
static class Iostream_init
```

This is supported on **z/OS**

Global Functions

__applFunc

```
bool
  __applFunc( void* element,
              void* applicator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

abs

```
double
  abs( complex z )
```

This is supported on **z/OS**

arg

```
double
  arg( complex z )
```

This is supported on **z/OS**

CLASS_MACRO

```
CLASS_MACRO()
```

This is supported on **OS/400 z/OS**

complex_zero

```
static const complex
  complex_zero( 0,
                0 )
```

This is supported on **z/OS**

conj

```
complex
  conj( complex a )
```

This is supported on **WIN OS/2 OS/400 z/OS**

conj

```
complex
  conj( const complex& a )
```

This is supported on **AIX**

copy

```
AType*
  copy( const AType& source )
```

Using the user provided copy constructor for the complete class, make a copy of the source object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

copyPointer

```
AType*
  copyPointer( const ICountedPointerTo < AType >& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

copyPointer

```
AType*
  copyPointer( const AType* source )
```

Using the user provided copy constructor for the complete class, make a copy of the source object.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

cos

```
complex
  cos( complex z )
```

This is supported on **z/OS**

cosh

```
complex
  cosh( complex z )
```

This is supported on **z/OS**

cosh

```
complex
  cosh( const complex& )
```

This is supported on **AIX**

createComparisonTest

```
IComparisonTestOf < AType >*
  createComparisonTest( AType* target,
                        AType* unequalTarget )
```

A global function which creates and returns an instance of IComparisonTestOf:

```
createComparisonTest(new ISample(1), new ISample(2));
```

which can be used with the adoptTest method (see the ITestCollection class).

This is supported on **WIN OS/2 AIX OS/400 z/OS**

createStreamTest

```
IStreamTestOf < AType >*
  createStreamTest( AType* target )
```

A global function which creates and returns an instance of IStreamTestOf:

```
createStreamTest(new ISample(2));
```

which can be used with the adoptTest method (see the ITestCollection class).

This is supported on **WIN OS/2 AIX OS/400 z/OS**

dec

```
ios&
  dec( ios& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

distance_type

```
ioc::numeric_t
  distance_type( const IFastTextIterator& )
```

Returns 0. distance_type isdefined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

distance_type

```
ioc::numeric_t
  distance_type( const ITextIterator& )
```

Returns 0. distance_type is defined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

distance_type

```
ioc::numeric_t
  distance_type( const IReverseIterator < ARealIterator,
                 AReturnType >& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

distance_type

```
ioc::numeric_t
  distance_type( const IConstTextIterator& )
```

Returns 0. distance_type isdefined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

elementForOps

```
Element&
  elementForOps( Element& e )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

elementForOps

```
Element const&
  elementForOps( Element const& e )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

endl

```
ITieredTextBuffer&
  endl( ITieredTextBuffer& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

endl

```
ostream&
  endl( ostream& i )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ends

```
ostream&
  ends( ostream& i )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

exp

```
complex
  exp( const complex& )
```

This is supported on **AIX**

exp

```
complex
  exp( complex z )
```

This is supported on **z/OS**

flush

```
ostream&
  flush( ostream& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

getline

```
std::istream&
  getline( std::istream& stream,
           IText& text,
           char delimiter = * IOCMSG ( ioc::kNEWLINE ) )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

getline

```
istream&
  getline( istream& stream,
           IText& text,
           char delimiter = * IOCMSG ( ioc::kNEWLINE ) )
```

Reads a ″line″ of characters from a stream into an IText. The original contents of the IText are replaced with the characters read from the stream, and characters are read from the stream until the stream is exhausted, the size of the IText reaches max_size(), or a character matching ″delimiter″ is read from the stream. The incoming characters are assumed to be in the host character set, and the default host-character-set-to-Unicode transcoder returned by ITranscoder::createTranscoder() is used.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

hash

```
ioc::unumeric_t
  hash( IInterest const& k,
        ioc::unumeric_t n )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

hex

```
ios&
  hex( ios& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

hexis

```
istream & IC_EXPORTI
  hexis( istream&,
         complex& )
```

This is supported on **z/OS**

hexos

```
ostream & IC_EXPORTI
  hexos( ostream&,
         complex )
```

This is supported on **z/OS**

ieeeis

```
istream & IC_EXPORTI
  ieeeis( istream&,
          complex& )
```

This is supported on **z/OS**

ieeeos

```
ostream & IC_EXPORTI
  ieeeos( ostream&,
          complex )
```

This is supported on **z/OS**

imag

```
double
  imag( const complex& a )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

key

```
const IThreadId&
  key( const INotificationProcessor* object )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

log

```
complex
  log( complex )
```

This is supported on **AIX**

log

```
complex
  log( complex z )
```

This is supported on **z/OS**

norm

```
double
  norm( complex z )
```

This is supported on **z/OS**

oct

```
ios&
  oct( ios& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !

```
bool
  operator !( const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !

```
int
  operator !( const IDecimalProxy& )
```

This is supported on **z/OS**

operator !=

```
bool
  operator !=( const ioc::unichar_t* a,
               const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const ILongLong&,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const char* pString1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const ILongLong&,
               unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IString& string1,
               char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const ILongLong&,
               long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const ILongLong&,
               unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
int
  operator !=( const IDecimalProxy&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator !=

```
bool
  operator !=( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( unsigned long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IText& a,
               const IText& b )
```

Returns true if the characters in a do not exactly match the characters in b. This is the same as IText(a).compare(b) != 0.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
int
  operator !=( int,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator !=

```
int
  operator !=( complex a,
               complex b )
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator !=

```
bool
  operator !=( unsigned int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
int
  operator !=( const IDecimalProxy&,
               int )
```

This is supported on **z/OS**

operator !=

```
int
  operator !=( const complex& a,
               const complex& b )
```

This is supported on **AIX**

operator !=

```
bool
  operator !=( const IString& string1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IThreadId& threadId1,
               const IThreadId& threadId2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator !=

```
bool
  operator !=( const IText& a,
               const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
IPair
  operator %( const IPair& aPair1,
              ioc::numeric_t divisor )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
IPair
  operator %( const IPair& pair1,
              const IPair& pair2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator %

```
ILongLong
  operator %( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
IString
  operator &( const char* pString,
              const IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &

```
ILongLong
  operator &( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( const ILongLong&,
               unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( const ILongLong&,
               unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( unsigned int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( unsigned long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( const ILongLong&,
               long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( const ILongLong&,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator &&

```
bool
  operator &&( int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IPair
  operator *( const IPair& pair1,
              const IPair& pair2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IDecimalResult
  operator *( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator *

```
ILongLong
  operator *( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IDecimalUtil
  operator *( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IDecimalResult
  operator *( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator *

```
IPair
  operator *( const IPair& pair1,
              double multiplier )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
complex
  operator *( complex a1,
              complex a2 )
```

This is supported on **z/OS**

operator *

```
ILongLong
  operator *( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
IDecimalResult
  operator *( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**

operator *

```
ILongLong
  operator *( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator *

```
ILongLong
  operator *( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ITextIterator
  operator +( ioc::numeric_t offset,
              const ITextIterator& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseIterator < ARealIterator , AReturnType >
  operator +( ioc::numeric_t offset,
              const IReverseIterator < ARealIterator,
              AReturnType >& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IString
  operator +( const char* pString,
              const IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalResult
  operator +( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator +

```
ILongLong
  operator +( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IText
  operator +( const IText& a,
              const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IConstTextIterator
  operator +( const IConstTextIterator& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IText
  operator +( const ioc::unichar_t* a,
              const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseTextIterator
  operator +( const IReverseTextIterator& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseConstTextIterator
  operator +( const IReverseConstTextIterator& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseIterator < ARealIterator , AReturnType >
  operator +( const IReverseIterator < ARealIterator,
              AReturnType >& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
complex
  operator +( const complex& a1,
              const complex& a2 )
```

This is supported on **AIX**

operator +

```
ILongLong
  operator +( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalResult
  operator +( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator +

```
IText
  operator +( const IText& a,
              ioc::unichar_t b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalResult
  operator +( const IDecimalProxy& )
```

This is supported on **z/OS**

operator +

```
IText
  operator +( const IText& a,
              const IText& b )
```

Concatenates two strings into a new IText. This is the same as IText(a) += b.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IFastTextIterator
  operator +( const IFastTextIterator& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseConstTextIterator
  operator +( ioc::numeric_t offset,
              const IReverseConstTextIterator& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IFastTextIterator
  operator +( ioc::numeric_t offset,
              const IFastTextIterator& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IText
  operator +( ioc::unichar_t a,
              const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IPair
  operator +( const IPair& pair1,
              const IPair& pair2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalResult
  operator +( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**

operator +

```
IConstTextIterator
  operator +( ioc::numeric_t offset,
              const IConstTextIterator& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IDecimalUtil
  operator +( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ILongLong
  operator +( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
ITextIterator
  operator +( const ITextIterator& iterator,
              ioc::numeric_t offset )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
IReverseTextIterator
  operator +( ioc::numeric_t offset,
              const IReverseTextIterator& iterator )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator +

```
complex
  operator +( complex a1,
              complex a2 )
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator -

```
IDecimalResult
  operator -( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator -

```
complex
  operator -( const complex& a )
```

This is supported on **AIX**

operator -

```
IPair
  operator -( const IPair& pair1,
              const IPair& pair2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
IDecimalResult
  operator -( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**

operator -

```
complex
  operator -( complex a1,
              complex a2 )
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator -

```
ILongLong
  operator -( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
complex
  operator -( const complex& a1,
              const complex& a2 )
```

This is supported on **AIX**

operator -

```
ILongLong
  operator -( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
IDecimalResult
  operator -( const IDecimalProxy& )
```

This is supported on **z/OS**

operator -

```
IDecimalResult
  operator -( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator -

```
ILongLong
  operator -( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
complex
  operator -( complex a )
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator -

```
ILongLong
  operator -( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
ILongLong
  operator -( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator -

```
IDecimalUtil
  operator -( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
ILongLong
  operator /( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
IDecimalResult
  operator /( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator /

```
complex
  operator /( complex a1,
              complex a2 )
```

This is supported on **z/OS**

operator /

```
ILongLong
  operator /( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
IPair
  operator /( const IPair& pair1,
              double divisor )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
ILongLong
  operator /( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
ILongLong
  operator /( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
ILongLong
  operator /( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
IPair
  operator /( const IPair& pair1,
              const IPair& pair2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
ILongLong
  operator /( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
ILongLong
  operator /( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
IDecimalResult
  operator /( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**
operator /

```
ILongLong
  operator /( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**
operator /

```
IDecimalUtil
  operator /( const IDecimalUtil& lv,
              const IDecimalUtil& rv )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
ILongLong
  operator /( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator /

```
IDecimalResult
  operator /( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator <

```
bool
  operator <( const ioc::unichar_t* a,
              const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IString& string1,
              const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
int
  operator <( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator <

```
bool
  operator <( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
int
  operator <( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator <

```
int
  operator <( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**

operator <

```
bool
  operator <( const IText& a,
              const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IString& string1,
              char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const char* pString1,
              const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IString& string1,
              const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <

```
bool
  operator <( const IText& a,
              const IText& b )
```

Returns true if a is lexically ″less than″ b. This is the same as IText(a).compare(b) == -1.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& stream,
               const IText& text )
```

Writes the characters in an IText out to an ostream. The Unicode character data is translated to the host character set using the default Unicode-to-host-character-set transocder returned by ITranscoder::createTranscoder().

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ILongLong
  operator <<( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& stream,
               const IRectangle& rectangle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IThreadHandle& aHandle )
```

This is supported on **WIN**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IProcessId& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IDate& aDate )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IPair& aRectangle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const ISharedSemaphoreHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IProcessId& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IThreadHandle& aHandle )
```

This is supported on **WIN**

operator <<

```
std::ostream&
  operator <<( std::ostream&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IModuleHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& s,
               complex a )
```

This is supported on **z/OS**

operator <<

```
ITieredTextBuffer&
  operator <<( ITieredTextBuffer&,
               ITest& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& os,
               const IBinaryCodedDecimal& bcd )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream&,
               complex )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& stream,
               const IRectangle& rectangle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IEnumHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const ITimeStamp& aTimeStamp )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IEnumHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& os,
               const IDecimalUtil& dut )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IThreadId& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const ISharedSemaphoreHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IPrivateSemaphoreHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
IStringParser
  operator <<( const IString& text,
               ioc::unumeric_t position )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& os,
               const IBinaryCodedDecimal& bcd )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const ITimeStamp& aTimeStamp )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& stream,
               const IText& text )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const IDate& aDate )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IPrivateSemaphoreHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream&
  operator <<( ostream& aStream,
               const IPair& aRectangle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const IThreadId& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& aStream,
               const ITime& aTime )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
ostream &
  operator <<( ostream& aStream,
               const ITime& aTime )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream &
  operator <<( std::ostream& os,
               const IDecimalUtil& dut )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<

```
std::ostream&
  operator <<( std::ostream& aStream,
               const IModuleHandle& aHandle )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( char& c,
                IDataStream& s )
```

Read a char value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( IStreamablePointer < Element >&,
                IDataStream& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( ICountedPointerTo < AType >&,
                IDataStream& fromStream )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( double& c,
                IDataStream& s )
```

Read a double value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( signed char& c,
                IDataStream& s )
```

Read a signed char value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( unsigned long& c,
                IDataStream& s )
```

Read a unsigned long value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( bool& c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( unsigned long long& c,
                IDataStream& s )
```

This is supported on **AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( long long& c,
                IDataStream& s )
```

This is supported on **AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( unsigned int& c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( IMStreamable& c,
                IDataStream& s )
```

Read a IMStreamable value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( long& c,
                IDataStream& s )
```

Read a long value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( unsigned char& c,
                IDataStream& s )
```

Read a unsigned char value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( int& c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( unsigned short& c,
                IDataStream& s )
```

Read a unsigned short value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( short& c,
                IDataStream& s )
```

Read a short value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <<=

```
void
  operator <<=( float& c,
                IDataStream& s )
```

Read a float value from an IDataStream. The value is returned to the caller by reference.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const IText& a,
               const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ILongLong&,
               long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const char* pString1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ILongLong&,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ILongLong&,
               unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
int
  operator <=( int,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator <=

```
bool
  operator <=( unsigned int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ILongLong&,
               unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const IString& string1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ioc::unichar_t* a,
               const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( unsigned long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
int
  operator <=( const IDecimalProxy&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator <=

```
bool
  operator <=( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
int
  operator <=( const IDecimalProxy&,
               int )
```

This is supported on **z/OS**

operator <=

```
bool
  operator <=( const IText& a,
               const IText& b )
```

Returns true if a is lexically ″less than or equal to″ b. This is the same as IText(a).compare(b) != 1.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator <=

```
bool
  operator <=( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const ioc::unichar_t* a,
               const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( unsigned int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const ILongLong&,
               long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const ILongLong&,
               unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const ILongLong&,
               unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
int
  operator ==( complex a,
               complex b )
```

This is supported on **WIN OS/2 OS/400 z/OS**

operator ==

```
bool
  operator ==( const ILongLong&,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
int
  operator ==( const IDecimalProxy&,
               int )
```

This is supported on **z/OS**

operator ==

```
int
  operator ==( int,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator ==

```
bool
  operator ==( const IString& string1,
               char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IText& a,
               const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IThreadId& threadId1,
               const IThreadId& threadId2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const char* pString1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IText& a,
               const IText& b )
```

Returns true if the characters in a exactly match the characters in b. This is the same as IText(a).compare(b) == 0.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
int
  operator ==( const complex& a,
               const complex& b )
```

This is supported on **AIX**

operator ==

```
bool
  operator ==( unsigned long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
bool
  operator ==( const IString& string1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ==

```
int
  operator ==( const IDecimalProxy&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator >

```
int
  operator >( const IDecimalProxy&,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator >

```
bool
  operator >( const IString& string1,
              const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const char* pString1,
              const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const IText& a,
              const IText& b )
```

Returns true if a is lexically ″greater than″ b. This is the same as IText(a).compare(b) == 1.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
int
  operator >( const IDecimalProxy&,
              int )
```

This is supported on **z/OS**

operator >

```
bool
  operator >( const ioc::unichar_t* a,
              const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const IString& string1,
              const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
int
  operator >( int,
              const IDecimalProxy& )
```

This is supported on **z/OS**

operator >

```
bool
  operator >( const IString& string1,
              const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >

```
bool
  operator >( const IText& a,
              const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( unsigned long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const ILongLong&,
               unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IText& a,
               const IText& b )
```

Returns true if a is lexically ″greater than or equal to″ b. This is the same as IText(a).compare(b) != -1.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IString& string1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( unsigned int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const ILongLong&,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
int
  operator >=( const IDecimalProxy&,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator >=

```
bool
  operator >=( const ILongLong&,
               unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IText& a,
               const ioc::unichar_t* b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const ioc::unichar_t* a,
               const IText& b )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( long,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( int,
               const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const ILongLong&,
               long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
bool
  operator >=( const IString& string1,
               const char* pString2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >=

```
int
  operator >=( const IDecimalProxy&,
               int )
```

This is supported on **z/OS**

operator >=

```
int
  operator >=( int,
               const IDecimalProxy& )
```

This is supported on **z/OS**

operator >=

```
bool
  operator >=( const char* pString1,
               const IString& string2 )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               IStringParser::Command command )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream&
  operator >>( std::istream& is,
               IBinaryCodedDecimal& bcd )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream&
  operator >>( istream&,
               IDecimal < w,
               p >& )
```

This is supported on **z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IStringTest& test )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IString& pattern )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               int position )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const IStringParser::SkipWords& skipObject )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream&
  operator >>( std::istream& stream,
               IText& text )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               const char* pattern )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream&
  operator >>( std::istream&,
               IDecimal < w,
               p >& )
```

This is supported on **z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               ioc::unumeric_t position )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream &
  operator >>( istream& aStream,
               IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream&
  operator >>( istream&,
               complex& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream&
  operator >>( istream& s,
               complex& a )
```

This is supported on **z/OS**

operator >>

```
ILongLong
  operator >>( const ILongLong&,
               int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               IString& token )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
IStringParser
  operator >>( const IString& text,
               char pattern )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream&
  operator >>( istream& stream,
               IText& text )
```

Reads characters from an istream into an IText. The original contents of the IText are replaced with the characters read from the stream, and characters are read from the stream until the stream is exhausted, the size of the IText reaches max_size(), or a whitespace character is detected. The incoming characters are assumed to be in the host character set, and the default host-character-set-to-Unicode transcoder returned by Transcoder::createTranscoder() is used.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
istream&
  operator >>( istream& is,
               IBinaryCodedDecimal& bcd )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>

```
std::istream &
  operator >>( std::istream& aStream,
               IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( long c,
                IDataStream& s )
```

Write a long value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( bool c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( signed char c,
                IDataStream& s )
```

Write a signed char value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( float c,
                IDataStream& s )
```

Write a float value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( const IMStreamable& object,
                IDataStream& s )
```

Write an IMStreamable object to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( const ICountedPointerTo < AType >&,
                IDataStream& toStream )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( unsigned short c,
                IDataStream& s )
```

Write a unsigned short value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( unsigned int c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( double c,
                IDataStream& s )
```

Write a double value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( unsigned char c,
                IDataStream& s )
```

Write a unsigned char value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( long long c,
                IDataStream& s )
```

This is supported on **AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( const IStreamablePointer < Element >&,
                IDataStream& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( unsigned long c,
                IDataStream& s )
```

Write a unsigned long value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( char c,
                IDataStream& s )
```

Write a char value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( short c,
                IDataStream& s )
```

Write a short value to an IDataStream.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( unsigned long long c,
                IDataStream& s )
```

This is supported on **AIX OS/400 z/OS**

operator >>=

```
void
  operator >>=( int c,
                IDataStream& s )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( const ILongLong&,
              int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( const ILongLong&,
              long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
IString
  operator ^( const char* pString,
              const IString& aString )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( unsigned long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( long,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( const ILongLong&,
              unsigned int )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( const ILongLong&,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( const ILongLong&,
              unsigned long )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

operator ^

```
ILongLong
  operator ^( unsigned int,
              const ILongLong& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

polar

```
complex
  polar( double,
         double )
```

This is supported on **AIX**

pow

```
complex
  pow( complex a,
       int n )
```

This is supported on **z/OS**

pow

```
complex
  pow( complex base,
       complex sup )
```

This is supported on **z/OS**

readAliasedObject

```
void
  readAliasedObject( AType *& theResult,
                     IDataStream& theStream )
```

Polymorphically read an object from an IDataStream. ″AType″ must descend from IMStreamable and IMRefCounted.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

readObject

```
void
  readObject( AType *& theResult,
              IDataStream& theStream )
```

Polymorphically read an object from an IDataStream. ″AType″ must descend from IMStreamable.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

real

```
double
  real( const complex& a )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

sin

```
complex
  sin( complex z )
```

This is supported on **z/OS**

sinh

```
complex
  sinh( const complex& )
```

This is supported on **AIX**

sinh

```
complex
  sinh( complex z )
```

This is supported on **z/OS**

sqrt

```
complex
  sqrt( complex z )
```

This is supported on **z/OS**

swap

```
void
  swap( IText& a,
        IText& b )
```

Exchanges the contents of two ITexts. This is the same as a.swap(b).

This is supported on **WIN OS/2 AIX OS/400 z/OS**

synchronizedInit

```
AType*
  synchronizedInit( ISynchronized < AType * >& var,
                    AType* value )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

transpose

```
IPair
  transpose( const IPair& aPair )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

value_type

```
ioc::unichar_t*
  value_type( const ITextIterator& )
```

Returns null. value_type is defined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

value_type

```
ioc::unichar_t*
  value_type( const IFastTextIterator& )
```

Returns null. value_type is defined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

value_type

```
ioc::unichar_t*
  value_type( const IConstTextIterator& )
```

Returns null. value_type is defined by STL, which uses it for overload resolution. The actual return value of the function is ignored.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

value_type

```
ioc::unichar_t*
  value_type( const IReverseIterator < ARealIterator,
              AReturnType >& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

writeAliasedObject

```
void
  writeAliasedObject( const AType* theObject,
                      IDataStream& theStream )
```

Polymorphically write an object to an IDataStream. "AType" must descend from IMStreamable and IMRefCounted.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

writeObject

```
void
  writeObject( const AType* theObject,
               IDataStream& theStream )
```

Polymorphically write an object to an IDataStream. "AType" must descend from IMStreamable.

This is supported on **WIN OS/2 AIX OS/400 z/OS**

ws

```
istream&
  ws( istream& )
```

This is supported on **WIN OS/2 AIX OS/400 z/OS**

Bibliography

For additional information about topics related to z/OS C/C++ programming and the IBM Open Class Library, refer to the following publications:

- *z/OS C/C++ Programming Guide*, SC09-4765
- *z/OS C/C++ User's Guide*, SC09-4767
- *C/C++ Language Reference*, SC09-4815
- *z/OS C/C++ Messages*, GC09-4819
- *z/OS C/C++ Run-Time Library Reference*, SA22-7821
- *z/OS C Curses*, SA22-7820
- *z/OS C/C++ Compiler and Run-Time Migration Guide*, GC09-4913
- *IBM Open Class Library User's Guide*, SC09-4811
- *Debug Tool User's Guide and Reference*, SC09-2137
- *Standard C++ Library Reference*, which is available at: `http://www.ibm.com/software/ad/c390/czos/czosdocs.html`

Related publications not listed in this section can be found on the IBM Online Library Omnibus Edition MVS Collection, SK2T-0710, the z/OS Collection, SK3T-4269, or on a tape available with z/OS.

© Copyright IBM Corp. 2001

Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created

© Copyright IBM Corp. 2001

programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Lab Director
IBM Canada Ltd. Laboratory
B3/KB7/8200/MKM
8200 Warden Avenue
Markham, Ontario, Canada L6G 1C7

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this information and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

Programming Interface Information

This publication is intended to help you develop applications that use the C or C++ classes provided with the IBM Open Class Library. This publication documents General-Use Programming Interface and Associated Guidance Information provided by the C and C++ classes.

General-Use programming interfaces allow the customer to write programs that obtain the services of the z/OS C/C++ compiler.

Trademarks and Service Marks

The following terms are trademarks of the International Business Machines Corporation in the United States or other countries or both:

AIX
BookManager
IBM
Language Environment
Open Class
OS/2
OS/390
OS/400
Presentation Manager
VisualAge
z/OS

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, Win32, and Windows are trademarks or registered trademarks of Microsoft Corporation in the United States, or other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, and service names may be the trademarks or service marks of others.

Industry Standards

The following standards are supported:

- The C language is consistent with the International Standard C (ANSI/ISO-IEC 9899–1990 [1992]). This standard has officially replaced American National standard for Information Systems-Programming Language C (X3.159–1989) and is technically equivalent to the ANSI C standard. The ILE C++ compiler supports the changes adopted into the C Standard by ISO/IEC 9899:1990/Amendment 1:1994.
- The C++ language is consistent with the International Standard for Information Systems-Programming Language C++ (ISO/IEC 14882:1998).

Program Number: 5694-A01

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC09-4812-01