



# NATURAL

---

**Natural**  
Dialog Components  
Version 5.1.1 for Windows



This document applies to Natural Version 5.1.1 for Windows and to all subsequent releases. Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

© June 2002, Software AG  
All rights reserved

Software AG and/or all Software AG products are either trademarks or registered trademarks of Software AG. Other products and company names mentioned herein may be the trademarks of their respective owners.

# Table of Contents

<b>Dialog Components - Overview</b>	1
Dialog Components - Overview	1
<b>Dialogs</b>	2
Dialogs	2
Description	2
Attributes for WINDOW Dialogs	3
Attributes for MDI Frame Window Dialogs	5
Attributes for MDI Child Window Dialogs	7
Events	8
<b>Dialog Elements</b>	9
Dialog Elements	9
ActiveX Controls	10
Description	10
Natural Attributes for ActiveX Controls	10
Events	11
Bitmap Control	12
Description	12
Attributes for Bitmap Control	12
Events	13
Canvas Control	14
Description	14
Attributes for Canvas Control	14
Events	15
Column-Specification Control	16
Description	16
Attributes for Column-Specification Control	16
COLUMN-TYPEs and their Attributes	17
Events	18
Context Menu	19
Description	19
Attributes for Context Menu	19
Events	19
Control Box Control	20
Description	20
Attributes for Control Box Control	20
Edit Area Control	21
Description	21
Attributes for Edit Area Control	21
Events	22
Font Control	23
Description	23
Attributes for Font Control	23
Events	23
Graphic-Text Control	24
Description	24
Attributes for Graphic-Text Control	24
Events	25
Group-Frame Control	26
Description	26
Attributes for Group-Frame Control	26
Events	27
GUI Control	28
Description	28

Attributes for GUI Control . . . . .	29
Input-Field Control . . . . .	33
Description . . . . .	33
Attributes for Input-Field Control . . . . .	33
Events . . . . .	34
Line Control . . . . .	35
Description . . . . .	35
Attributes for Line Control . . . . .	35
Events . . . . .	36
List-Box Control . . . . .	37
Description . . . . .	37
Attributes for List-Box Control . . . . .	37
Events . . . . .	38
List-Box Item . . . . .	39
Description . . . . .	39
Attributes for List-Box Item . . . . .	39
Events . . . . .	40
Menu Bar . . . . .	41
Description . . . . .	41
Attributes for Menu Bar . . . . .	41
Events . . . . .	42
Menu Item . . . . .	43
Description . . . . .	43
Attributes for Menu Item . . . . .	43
Events . . . . .	45
OLE Container Control . . . . .	46
Description . . . . .	46
Attributes for OLE Container Control . . . . .	46
Events . . . . .	47
Push-Button Control . . . . .	48
Description . . . . .	48
Attributes for Push-Button Control . . . . .	48
Events . . . . .	49
Radio-Button Control . . . . .	50
Description . . . . .	50
Attributes for Radio-Button Control . . . . .	50
Events . . . . .	52
Rectangle Control . . . . .	53
Description . . . . .	53
Attributes for Rectangle Control . . . . .	53
Events . . . . .	54
Scrollbar Control . . . . .	55
Description . . . . .	55
Attributes for Scrollbar Control . . . . .	55
Events . . . . .	56
Selection-Box Control . . . . .	57
Description . . . . .	57
Attributes for Selection-Box Control . . . . .	57
Events . . . . .	58
Selection-Box Item . . . . .	60
Description . . . . .	60
Attributes for Selection-Box Item . . . . .	60
Events . . . . .	61
Signal . . . . .	62
Description . . . . .	62
Attributes for Signal . . . . .	62

Events . . . . .	63
Status-Bar Control . . . . .	64
Description . . . . .	64
Attributes for Status-Bar Control . . . . .	64
Events . . . . .	65
Status-Bar Pane . . . . .	66
Description . . . . .	66
Attributes for Status-Bar Pane . . . . .	66
Events . . . . .	67
Submenu Control . . . . .	68
Description . . . . .	68
Attributes for Submenu Control . . . . .	68
Events . . . . .	69
Table Control . . . . .	70
Description . . . . .	70
Attributes for Table Control . . . . .	70
Attributes for Cells in a Table Control . . . . .	72
COLUMN-TYPEs in Cells and Their Attributes . . . . .	72
Events . . . . .	72
Text-Constant Control . . . . .	74
Description . . . . .	74
Attributes for Text-Constant Control . . . . .	74
Events . . . . .	75
Timer . . . . .	76
Description . . . . .	76
Attributes for Timer . . . . .	76
Events . . . . .	76
Toggle-Button Control . . . . .	77
Description . . . . .	77
Attributes for Toggle-Button Control . . . . .	77
Events . . . . .	78
Tool Bar . . . . .	79
Description . . . . .	79
Attributes for Tool Bar . . . . .	79
Events . . . . .	80
Tool-Bar Control . . . . .	81
Description . . . . .	81
Attributes for Tool-Bar Control . . . . .	81
Events . . . . .	82
Tool-Bar Item . . . . .	83
Description . . . . .	83
Attributes for Tool-Bar Item . . . . .	83
Events . . . . .	84
<b>Attributes . . . . .</b>	<b>85</b>
Attributes . . . . .	85
Alphabetical List of Attributes . . . . .	85
ACCELERATOR . . . . .	87
ACTIVE-CHILD . . . . .	89
AUTOADJUST . . . . .	90
AUTOSELECT . . . . .	91
BACKGROUND-COLOUR-NAME . . . . .	92
BACKGROUND-COLOUR-VALUE . . . . .	93
BAR-ID . . . . .	94
BITMAP-FILE-NAME . . . . .	95
CELL-ATTRIBUTES . . . . .	96
CHECKED . . . . .	96

CLIENT-DATA . . . . .	97
CLIENT-HANDLE . . . . .	98
CLIENT-KEY . . . . .	99
CLIENT-VALUE . . . . .	100
COLUMN . . . . .	101
COLUMN-COUNT . . . . .	102
COLUMN-TYPE . . . . .	103
CONTEXT-MENU . . . . .	104
DEFAULT-BUTTON . . . . .	105
DIL-TEXT . . . . .	106
DOCKING . . . . .	107
DPI . . . . .	108
DRAGGABLE . . . . .	109
EDIT-MASK . . . . .	110
EMBEDDED-OBJECT . . . . .	111
ENABLED . . . . .	112
EVENT-QUEUEING . . . . .	113
FIRST-CHILD . . . . .	114
FIRST-COLUMN-WIDTH . . . . .	115
FIRST-VISIBLE-COLUMN . . . . .	116
FIRST-VISIBLE-ROW . . . . .	117
FOLLOWS . . . . .	118
FONT-HANDLE . . . . .	119
FONT-STRING . . . . .	120
FOREGROUND-COLOUR-NAME . . . . .	122
FOREGROUND-COLOUR-VALUE . . . . .	123
FROZEN-COLUMNS . . . . .	124
GROUP-ID . . . . .	125
HANDLE-VARIABLE . . . . .	126
HAS-DIL . . . . .	127
HAS-FIRST-COLUMN . . . . .	128
HAS-HELP-BUTTON . . . . .	129
HAS-MENU-BAR . . . . .	130
HAS-STATUS-BAR . . . . .	131
HAS-SYSTEM-BUTTON . . . . .	132
HAS-TOOLBAR . . . . .	133
HAS-TOOLTIP . . . . .	134
HEADER-FONT-HANDLE . . . . .	135
HEADER-HEIGHT . . . . .	136
HELP-FILENAME . . . . .	137
HELP-ID . . . . .	138
HORIZ-SCROLLABLE . . . . .	139
ICONIZED . . . . .	140
INPLACE-ACTIVE . . . . .	141
ITEM-H . . . . .	142
ITEM-W . . . . .	143
LAST-CHILD . . . . .	144
LENGTH . . . . .	145
LINE . . . . .	146
LINKED . . . . .	147
LOCATION . . . . .	148
MARGIN-X . . . . .	149
MARGIN-Y . . . . .	150
MAX . . . . .	151
MAXIMIZABLE . . . . .	152
MAXIMIZED . . . . .	153

MENU-HANDLE . . . . .	154
MENU-ITEM-OLE . . . . .	155
MENU-ITEM-TYPE . . . . .	156
MIN . . . . .	158
MINIMIZABLE . . . . .	159
MINIMIZED . . . . .	160
MODIFIABLE . . . . .	161
MODIFIED . . . . .	162
MODIFIED-SUCCESSOR . . . . .	163
MULTI-SELECTION . . . . .	164
NAME . . . . .	165
OBJECT-SIZE . . . . .	166
OFFSET-X . . . . .	167
OFFSET-Y . . . . .	168
P1-X . . . . .	169
P1-Y . . . . .	170
P2-X . . . . .	171
P2-Y . . . . .	172
PAGE . . . . .	173
PARENT . . . . .	174
POPUP-HELP . . . . .	175
PREDECESSOR . . . . .	176
RECTANGLE-H . . . . .	177
RECTANGLE-W . . . . .	178
RECTANGLE-X . . . . .	179
RECTANGLE-Y . . . . .	180
ROW . . . . .	181
ROW-COUNT . . . . .	182
ROW-HEIGHTx . . . . .	183
SAME-AS . . . . .	184
SCROLLRANGE-X . . . . .	186
SCROLLRANGE-Y . . . . .	187
SELECTED . . . . .	188
SELECTED-SUCCESSOR . . . . .	189
SERVER-OBJECT . . . . .	190
SERVER-PROGID . . . . .	191
SHARED . . . . .	192
SIZE-MODIFIABLE . . . . .	193
SLIDER . . . . .	194
SORTED . . . . .	195
STATUS-HANDLE . . . . .	196
STATUS-TEXT . . . . .	197
STRING . . . . .	198
STYLE . . . . .	199
SUCCESSOR . . . . .	205
SUPPRESS-ACTIVATE-EVENT . . . . .	206
SUPPRESS-BEFORE-OPEN-EVENT . . . . .	207
SUPPRESS-CHANGE-EVENT . . . . .	208
SUPPRESS-CLICK-EVENT . . . . .	209
SUPPRESS-CLIENT-SIZE-EVENT . . . . .	210
SUPPRESS-CLOSE-EVENT . . . . .	211
SUPPRESS-DBL-CLICK-EVENT . . . . .	212
SUPPRESS-DELETE-ROW-EVENT . . . . .	213
SUPPRESS-ENTER-EVENT . . . . .	214
SUPPRESS-ENTER-CELL-EVENT . . . . .	215
SUPPRESS-FILL-EVENT . . . . .	216

SUPPRESS-IDLE-EVENT . . . . .	217
SUPPRESS-INSERT-ROW-EVENT . . . . .	218
SUPPRESS-LEAVE-EVENT . . . . .	219
SUPPRESS-LEAVE-CELL-EVENT . . . . .	220
SUPPRESS-SIZE-EVENT . . . . .	221
SUPPRESS-COMMAND-STATUS-EVENT . . . . .	222
SUPPRESS-TOP-EVENT . . . . .	223
TIMER-INTERVAL . . . . .	224
TOOLBAR-HANDLE . . . . .	225
TOOLBAR-POS . . . . .	226
TOOLTIP . . . . .	227
TYPE . . . . .	228
VARIABLE . . . . .	230
VERSION . . . . .	231
VERT-SCROLLABLE . . . . .	232
VISIBLE . . . . .	233
ZOOM-FACTOR . . . . .	234
<b>Events . . . . .</b>	<b>235</b>
Events . . . . .	235
Activate Event . . . . .	236
Applies To . . . . .	236
Description . . . . .	236
After-Any Event . . . . .	236
Applies To . . . . .	236
Description . . . . .	236
After-Open Event . . . . .	236
Applies To . . . . .	236
Description . . . . .	236
Before-Any Event . . . . .	237
Applies To . . . . .	237
Description . . . . .	237
Before-Open Event . . . . .	237
Applies To . . . . .	237
Description . . . . .	237
Change Event . . . . .	238
Applies To . . . . .	238
Description . . . . .	238
Click Event . . . . .	238
Applies To . . . . .	238
Description . . . . .	238
Client-Size Event . . . . .	239
Applies To . . . . .	239
Description . . . . .	239
Close Event . . . . .	240
Applies To . . . . .	240
Description . . . . .	240
Command-Status Event . . . . .	240
Applies To . . . . .	240
Description . . . . .	240
DDE-Client Event . . . . .	241
Applies To . . . . .	241
Description . . . . .	241
DDE-Server Event . . . . .	242
Applies To . . . . .	242
Description . . . . .	242
Default Event . . . . .	242

Applies To . . . . .	242
Description . . . . .	242
Delete-Row Event . . . . .	243
Applies To . . . . .	243
Description . . . . .	243
Double-Click Event . . . . .	243
Applies To . . . . .	243
Description . . . . .	243
Drag-Drop Event . . . . .	244
Applies To . . . . .	244
Description . . . . .	244
Enter Event . . . . .	244
Applies To . . . . .	244
Description . . . . .	244
Enter-Cell Event . . . . .	245
Applies To . . . . .	245
Description . . . . .	245
Error Event . . . . .	245
Applies To . . . . .	245
Description . . . . .	245
Fill Event . . . . .	246
Applies To . . . . .	246
Description . . . . .	246
Idle Event . . . . .	246
Applies To . . . . .	246
Description . . . . .	246
Insert-Row Event . . . . .	246
Applies To . . . . .	246
Description . . . . .	246
Leave Event . . . . .	247
Applies To . . . . .	247
Description . . . . .	247
Leave-Cell Event . . . . .	248
Applies To . . . . .	248
Description . . . . .	248
Size Event . . . . .	248
Applies To . . . . .	248
Description . . . . .	248
Top Event . . . . .	248
Applies To . . . . .	248
Description . . . . .	248
User-Defined Events . . . . .	249
Apply To . . . . .	249
Description . . . . .	249
<b>PROCESS GUI Statement Actions . . . . .</b>	<b>250</b>
PROCESS GUI Statement Actions . . . . .	250
General Information . . . . .	250
ADD Action . . . . .	250
Description . . . . .	250
Parameters for the ADD WITH option . . . . .	251
ADD-ITEMS Action . . . . .	253
Description . . . . .	253
Parameters . . . . .	253
ADD-ITEMS-EX Action . . . . .	254
Description . . . . .	254
Parameters . . . . .	254

BEEP Action . . . . .	256
Description . . . . .	256
Parameters . . . . .	256
CLEAR Action . . . . .	257
Description . . . . .	257
Parameters . . . . .	257
DELETE Action . . . . .	258
Description . . . . .	258
Parameters . . . . .	258
DELETE-CHILDREN Action . . . . .	259
Description . . . . .	259
Parameters . . . . .	259
DELETE-WINDOW Action . . . . .	260
Description . . . . .	260
Parameters . . . . .	260
EDIT-GET-LINE-NUMBER Action . . . . .	261
Description . . . . .	261
Parameters . . . . .	261
EDIT-LINE-DELETE Action . . . . .	262
Description . . . . .	262
Parameters . . . . .	262
EDIT-LINE-GET-SELECTION Action . . . . .	263
Description . . . . .	263
Parameters . . . . .	263
EDIT-LINE-GET-TEXT Action . . . . .	264
Description . . . . .	264
Parameters . . . . .	264
EDIT-LINE-INSERT Action . . . . .	265
Description . . . . .	265
Parameters . . . . .	265
EDIT-LINE-SET-SELECTION Action . . . . .	266
Description . . . . .	266
Parameters . . . . .	266
EDIT-LINE-SET-TEXT Action . . . . .	267
Description . . . . .	267
Parameters . . . . .	267
GET-FOCUS Action . . . . .	268
Description . . . . .	268
Parameters . . . . .	268
GET-MESSAGE-TEXT Action . . . . .	269
Description . . . . .	269
Parameters . . . . .	269
HELP Action . . . . .	270
Description . . . . .	270
Parameters . . . . .	270
HOURGLASS-REMOVE Action . . . . .	271
Description . . . . .	271
Parameters . . . . .	271
HOURGLASS-STACK Action . . . . .	272
Description . . . . .	272
Parameters . . . . .	272
HOURGLASS-UNSTACK Action . . . . .	273
Description . . . . .	273
Parameters . . . . .	273
INPUT-COPY-SELECTION Action . . . . .	274
Description . . . . .	274

Parameters . . . . .	274
INPUT-CUT-SELECTION Action . . . . .	275
Description . . . . .	275
Parameters . . . . .	275
INPUT-DELETE-SELECTION Action . . . . .	276
Description . . . . .	276
Parameters . . . . .	276
INPUT-GET-LINE-LENGTH Action . . . . .	277
Description . . . . .	277
Parameters . . . . .	277
INPUT-GET-SELECTION Action . . . . .	278
Description . . . . .	278
Parameters . . . . .	278
INPUT-GET-TEXT Action . . . . .	279
Description . . . . .	279
Parameters . . . . .	279
INPUT-PASTE Action . . . . .	280
Description . . . . .	280
Parameters . . . . .	280
INPUT-SET-SELECTION Action . . . . .	281
Description . . . . .	281
Parameters . . . . .	281
INPUT-SET-TEXT Action . . . . .	282
Description . . . . .	282
Parameters . . . . .	282
INPUT-UNDO Action . . . . .	283
Description . . . . .	283
Parameters . . . . .	283
INQ-CLICKPOSITION Action . . . . .	284
Description . . . . .	284
Parameters . . . . .	284
INQ-DRAG-DROP Action . . . . .	285
Description . . . . .	285
Parameters . . . . .	285
INQ-INNER-RECT Action . . . . .	286
Description . . . . .	286
Parameters . . . . .	286
INQ-ITEM-BY-POSITION Action . . . . .	287
Description . . . . .	287
Parameters . . . . .	287
LOAD-LAYOUT Action . . . . .	288
Description . . . . .	288
Parameters . . . . .	288
MOVE-NAVIGATION-ITEMS Action . . . . .	289
Description . . . . .	289
Parameters . . . . .	289
OLE-ACTIVATE . . . . .	290
Description . . . . .	290
Parameters . . . . .	290
OLE-DEACTIVATE . . . . .	291
Description . . . . .	291
Parameters . . . . .	291
OLE-GET-DATA . . . . .	292
Description . . . . .	292
Parameters . . . . .	292
OLE-INSERT-OBJECT . . . . .	293

Description . . . . .	293
Parameters . . . . .	293
OLE-READ-FROM-FILE . . . . .	294
Description . . . . .	294
Parameters . . . . .	294
OLE-SAVE-TO-FILE . . . . .	295
Description . . . . .	295
Parameters . . . . .	295
OLE-SET-DATA . . . . .	296
Description . . . . .	296
Parameters . . . . .	296
PICK-FILENAME Action . . . . .	297
Description . . . . .	297
Parameters . . . . .	297
PLAY-SOUND Action . . . . .	299
Description . . . . .	299
Parameters . . . . .	299
PROCESS-EVENTS Action . . . . .	300
Description . . . . .	300
Parameters . . . . .	300
RECALC-LAYOUT Action . . . . .	301
Description . . . . .	301
Parameters . . . . .	301
REFRESH-LINKS Action . . . . .	302
Description . . . . .	302
Parameters . . . . .	302
RESET-ATTRIBUTES Action . . . . .	303
Description . . . . .	303
Parameters . . . . .	303
SAVE-LAYOUT Action . . . . .	304
Description . . . . .	304
Parameters . . . . .	304
SET-FOCUS Action . . . . .	304
Description . . . . .	304
Parameters . . . . .	304
SET-TABS Action . . . . .	306
Description . . . . .	306
Parameters . . . . .	306
SYSTEM-GET-NATIVE-HANDLE Action . . . . .	307
Description . . . . .	307
Parameters . . . . .	307
SYSTEM-PRINTER-SETUP Action . . . . .	308
Description . . . . .	308
Parameters . . . . .	308
TABLE-DELETE-ROW Action . . . . .	309
Description . . . . .	309
Parameters . . . . .	309
TABLE-FIND-FIELD Action . . . . .	310
Description . . . . .	310
Parameters . . . . .	310
TABLE-GET-SELECTION Action . . . . .	311
Description . . . . .	311
Parameters . . . . .	311
TABLE-INQUIRE-CELL Action . . . . .	312
Description . . . . .	312
Parameters . . . . .	312

TABLE-INQUIRE-ROW Action . . . . .	313
Description . . . . .	313
Parameters . . . . .	313
TABLE-INSERT-ROW Action . . . . .	314
Description . . . . .	314
Parameters . . . . .	314
TABLE-REFRESH Action . . . . .	315
Description . . . . .	315
Parameters . . . . .	315
TABLE-SET-SELECTION Action . . . . .	316
Description . . . . .	316
Parameters . . . . .	316
TEXT-GET-EXTENT Action . . . . .	317
Description . . . . .	317
Parameters . . . . .	317
UPDATE-COMMAND-STATUS Action . . . . .	318
Description . . . . .	318
Parameters . . . . .	318
VALIDATE Action . . . . .	319
Description . . . . .	319
Parameters . . . . .	319
<b>NGU Subprograms and Dialogs . . . . .</b>	<b>320</b>
NGU Subprograms and Dialogs . . . . .	320
General Information . . . . .	320
NGULKEY1 Reserved Symbols . . . . .	320
Color Symbols . . . . .	320
Dialog Element Types . . . . .	321
Event-Suppressing Symbols . . . . .	321
Menu Item Style Symbols . . . . .	321
Menu Item Symbols . . . . .	322
Separator Symbols . . . . .	323
Tool Bar Symbols . . . . .	323
NGU-CLIENT-ADVISE-HOT Subprogram . . . . .	324
Natural Object Name . . . . .	324
Description . . . . .	324
Parameters . . . . .	324
NGU-CLIENT-ADVISE-TERM Subprogram . . . . .	325
Natural Object Name . . . . .	325
Description . . . . .	325
Parameters . . . . .	325
NGU-CLIENT-ADVISE-WARM Subprogram . . . . .	326
Natural Object Name . . . . .	326
Description . . . . .	326
Parameters . . . . .	326
NGU-CLIENT-CONNECT Subprogram . . . . .	327
Natural Object Name . . . . .	327
Description . . . . .	327
Parameters . . . . .	327
NGU-CLIENT-DISCONNECT Subprogram . . . . .	328
Natural Object Name . . . . .	328
Description . . . . .	328
Parameters . . . . .	328
NGU-CLIENT-EXECUTE Subprogram . . . . .	329
Natural Object Name . . . . .	329
Description . . . . .	329
Parameters . . . . .	329

NGU-CLIENT-GET-DATA Subprogram . . . . .	330
Natural Object Name . . . . .	330
Description . . . . .	330
Parameters . . . . .	331
NGU-CLIENT-POKE Subprogram . . . . .	332
Natural Object Name . . . . .	332
Description . . . . .	332
Parameters . . . . .	332
NGU-CLIENT-REQUEST Subprogram . . . . .	333
Natural Object Name . . . . .	333
Description . . . . .	333
Parameters . . . . .	333
NGU-CLIENT-STOP Subprogram . . . . .	334
Natural Object Name . . . . .	334
Description . . . . .	334
Parameters . . . . .	334
NGU-COLOUR-SELECT Dialog . . . . .	335
Natural Object Name . . . . .	335
Description . . . . .	335
OPEN DIALOG Parameters . . . . .	335
NGU-DIALOG-CLOSE-ALL Subprogram and Subroutine . . . . .	336
Natural Object Names . . . . .	336
Description . . . . .	336
Parameters . . . . .	336
NGU-FONT-SELECT Dialog . . . . .	337
Natural Object Name . . . . .	337
Description . . . . .	337
OPEN DIALOG Parameters . . . . .	337
NGU-MESSAGEBOX Dialog . . . . .	338
Natural Object Name . . . . .	338
Description . . . . .	338
OPEN DIALOG Parameters . . . . .	338
Separator Keyword . . . . .	340
Usage in Enter or Leave Event Handlers . . . . .	340
NGU-SERVER-DATA Subprogram . . . . .	341
Natural Object Name . . . . .	341
Description . . . . .	341
Parameters . . . . .	341
NGU-SERVER-GET-DATA Subprogram . . . . .	342
Natural Object Name . . . . .	342
Description . . . . .	342
Parameters . . . . .	343
NGU-SERVER-WAIT Subprogram . . . . .	344
Natural Object Name . . . . .	344
Description . . . . .	344
Parameters . . . . .	345
NGU-SERVER-REGISTER Subprogram . . . . .	346
Natural Object Name . . . . .	346
Description . . . . .	346
Parameters . . . . .	346
NGU-SERVER-STOP Subprogram . . . . .	347
Natural Object Name . . . . .	347
Description . . . . .	347
Parameters . . . . .	347
NGU-SERVER-UNREGISTER Subprogram . . . . .	348
Natural Object Name . . . . .	348

Description . . . . .	348
Parameters . . . . .	348
<b>Enhanced Source Code Format . . . . .</b>	<b>349</b>
Enhanced Source Code Format . . . . .	349
Syntax Conventions . . . . .	349
How Natural Dialogs Work . . . . .	350
Syntax . . . . .	351
General Syntax . . . . .	351
Subsections of the General Syntax . . . . .	352
Subordinate Syntax Sections . . . . .	358



# Dialog Components - Overview

The Natural Dialog Components documentation provides a reference of the components that can be used when developing an event-driven application with the dialog editor. These components include:

- Dialogs
- Dialog Elements
- Attributes
- Events
- PROCESS GUI Statement Actions
- NGU Subprograms and Dialogs
- The Enhanced Source Code Format

When working with the Natural dialog editor, you should have a working knowledge of Microsoft Windows and its terminology. If not, consult the Windows documentation for a description of basic Windows elements, usage and terminology.

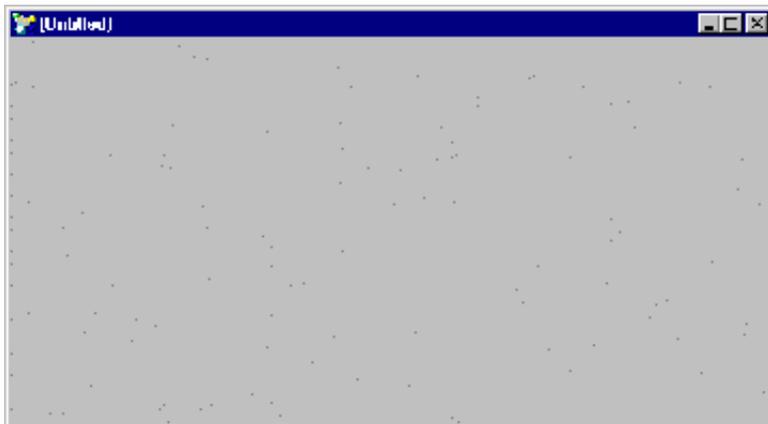
# Dialogs

The following topics are covered below:

- Description
  - Attributes for WINDOW Dialogs
  - Attributes for MDI Frame Window Dialogs
  - Attributes for MDI Child Window Dialogs
  - Events
- 

## Description

A dialog represents a window of an event-driven application. It is a Natural object like a map or a program. You create, save, stow and run it with the dialog editor.



A dialog is available as the following window types (values of the attribute TYPE):

- (Standard) window
- MDI frame window
- MDI child window

## Attributes for WINDOW Dialogs

Attribute Name	Query	Set/Modify	In Attr. Window
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CONTEXT-MENU	X	X/X	X
CLIENT-VALUE	X	X/X	
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-HELP-BUTTON	X	X/-	X
HAS-MENU-BAR	X	X/-	
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	X/-	X
HAS-TOOLBAR	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	X/-	X
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	X/-	X
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	X
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TOOLBAR-POS	X	X/-	X
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

## Attributes for MDI Frame Window Dialogs

Attribute Name	Query	Set/Modify	In Attr. Window
ACTIVE-CHILD	X	-/X	
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CONTEXT-MENU	X	X/X	X
CLIENT-VALUE	X	X/X	
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
HAS-HELP-BUTTON	X	X/-	X
HAS-MENU-BAR	X	X/-	
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	X/-	X
HAS-TOOLBAR	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	-/-	X
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	X/-	X
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
MODIFIED-SUCCESSOR	X	-/-	
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	X
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TOOLBAR-POS	X	X/-	X
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

## Attributes for MDI Child Window Dialogs

Attribute Name	Query	Set/Modify	In Attr. Window
AUTO-ADJUST	X	-/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/-	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DEFAULT-BUTTON	X	X/X	X
DOCKING	X	X/X	X
DPI	X	X/-	
ENABLED	X	X/X	X
EVENT-QUEUEING	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	-/-	
FONT-STRING	X	X/-	X
HAS-DIL	X	X/X	X
HAS-HELP-BUTTON	X	X/-	X
HAS-MENU-BAR	X	X/-	
HAS-STATUS-BAR	X	X/X	X
HAS-SYSTEM-BUTTON	X	-/-	
HAS-TOOLBAR	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
MAXIMIZABLE	X	-/-	
MAXIMIZED	X	X/X	X
MENU-HANDLE	X	X/X	
MINIMIZABLE	X	-/-	
MINIMIZED	X	X/X	X
MODIFIED	X	X/X	
MODIFIED-SUCCESSOR	X	-/-	

Attribute Name	Query	Set/Modify	In Attr. Window
NAME	X	-/-	
PARENT	X	X/-	
POPUP-HELP	X	X/X	X
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SCROLLRANGE-X	X	X/X	
SCROLLRANGE-Y	X	X/X	
SIZE-MODIFIABLE	X	-/-	
STATUS-HANDLE	X	X/X	
STATUS-TEXT		-/X	
STRING	X	X/X	X
STYLE	X	-/X	X
SUCCESSOR	X	-/-	
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-IDLE-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-SIZE-EVENT	X	X/X	
TOOLBAR-HANDLE	X	X/-	
TYPE	X	X/-	X
VERSION	X	-/-	
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

## Events

After-open event, before-close event, before-open event, command-status-event (may be suppressed), enter event (may be suppressed), error event, idle event (may be suppressed), leave event (may be suppressed), size event (may be suppressed).

# Dialog Elements

---

## ActiveX Controls



This is an example of an ActiveX control.

### Description

ActiveX controls are third-party custom controls that you can integrate into a Natural dialog. In addition to the Natural attributes listed below, the properties, methods and events of the respective ActiveX control are available. The handling of these properties, methods and events is described in the User's Guide - Event driven programming techniques - Working with ActiveX controls..

### Natural Attributes for ActiveX Controls

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

This dialog element does not create Natural-specific events but generates Natural events from the events available with the ActiveX control.

## Bitmap Control



### Description

A bitmap control is a picture to be displayed anywhere within a dialog window. It helps explain the purpose of dialog elements in a graphical way. The end user may click upon a bitmap control and drag it onto another bitmap control in the same dialog. You can allow this by using the attributes `ENABLED` and `DRAGGABLE` and the `PROCESS GUI` statement actions `INQ-CLICKPOSITION` and `INQ-DRAG-DROP`.

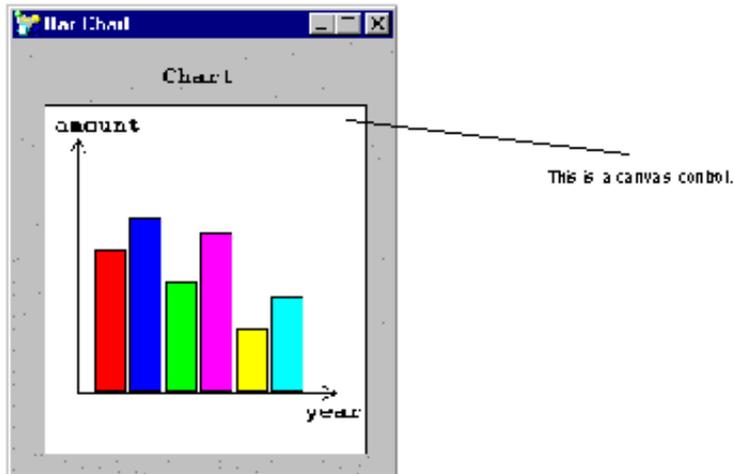
### Attributes for Bitmap Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
DRAGGABLE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event (may be suppressed), double-click event (may be suppressed), drag and drop event.

## Canvas Control



### Description

A canvas control provides a necessary background for the rectangle, line and graphic-text controls. Once you have created a canvas control in the dialog, you can go on to create the rectangle, line and graphic-text controls in it. The rectangle, line and graphic-text controls are then displayed inside the borders of the canvas control; if they go beyond the canvas borders, they are clipped.

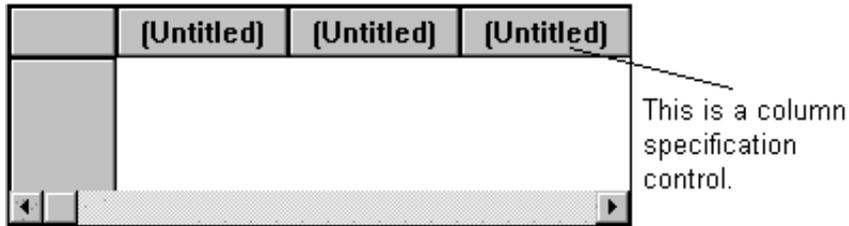
### Attributes for Canvas Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
OFFSET-X	X	X/X	
OFFSET-Y	X	X/X	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event (may be suppressed), double-click event (may be suppressed).

## Column-Specification Control



### Description

A column-specification control is a dialog element that defines the columns in a table control. (A table control is a dialog element that represents a spreadsheet.) Once the table control defines the spreadsheet as such, its columns are defined by adding column-specification controls.

### Attributes for Column-Specification Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COLUMN-TYPE	X	X/-	X
DIL-TEXT	X	X/X	X
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/-	X
MODIFIABLE (input-field)	X	X/X	X
MODIFIABLE (selection-box)	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-W	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-ENTER-CELL-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-LEAVE-CELL-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	-/-	

## COLUMN-TYPEs and their Attributes

The attributes that are valid for column specification controls in general are not always available depending on the value of the attribute COLUMN-TYPE. Such a value might be, for example, "toggle-button control". The column then consists of cells that can be used like toggle-button controls. For these toggle-button cells, only a subset of the attributes is available.

The following table specifies which attributes are *not* applicable to a COLUMN-TYPE.

<b>COLUMN-TYPE</b>	<b>Attributes NOT Available</b>
Input field control	(All available.)
Selection box control	(All available.)
Selection box item	DIL-TEXT, HELP-ID, LENGTH, MODIFIABLE, STYLE, SUPPRESS-CHANGE-EVENT, SUPPRESS-CLICK-EVENT, SUPPRESS-DBL-CLICK-EVENT, SUPPRESS-ENTER-EVENT, SUPPRESS-ENTER-CELL-EVENT, SUPPRESS-LEAVE-EVENT, SUPPRESS-LEAVE-CELL-EVENT, VISIBLE.
Toggle button control	LENGTH, STYLE, MODIFIABLE.

## Events

This dialog element does not create events.

## Context Menu

### Description

The context menu allows you to define menus which appear when the user clicks a dialog or dialog element with the right mouse button. You can retrieve the position of the click via the INQ-CLICKPOSITION action, should this be necessary.

Context menus are defined separately from the dialog elements and associated with the relevant dialog element(s) via the CONTEXT-MENU attribute. The context menu can be modified before it is displayed via the before-open event.

The context menu is constructed and handled almost identically to the submenu control. Like the submenu control, a context menu can contain menu items which open up submenus when selected. Each submenu in such a multi-level context menu receives its own before-open event each time the submenu is displayed.

### Attributes for Context Menu

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/-	
SUCCESSOR	X	-/-	
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X	
TYPE	X	X/-	

### Events

Before-open event (may be suppressed).

## Control Box Control

### Description

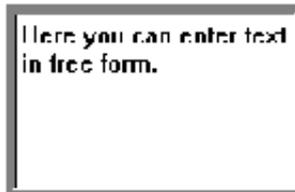
The control box control is a general-purpose container control. Any dialog elements placed within the control box become child controls of the control box, thus allowing related dialog elements to be grouped together for programming convenience or ease of user manipulation in the dialog editor. For example, when a control box is made invisible, all the dialog elements it contains become invisible, and when it is moved, these dialog elements are moved with it.

Special attention has been paid to making it possible to create multiple "pages" of dialog elements at the same position in a dialog using control boxes marked with the 'Exclusive' style. Only one such page can then be visible at any one time, both at edit-time and at run-time, and the Dialog Editor automatically changes the active page according to the current selection. This feature can be used to support wizard dialogs and ActiveX tab controls. Please refer to the article Working with Control Boxes for more information.

### Attributes for Control Box Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Edit Area Control



### Description

In an edit area control, the end user may type in free-form text. It may contain any number of lines, and scroll bars may be set: if there are more lines in the edit area control than can be displayed, the end user may scroll to the desired line.

To use the free form text elsewhere in your application, you will have to set the text of the edit area control into alphanumeric Natural fields line by line. You can then query these lines of text one after the other.

You can manipulate the text in the edit area control by using PROCESS GUI statement actions. This helps you insert a new line, query a certain line, set a selection, query the selected text, and so on. Note that for moving text from one edit area control to another, the STRING attribute will only be able to hold 253 characters. If you want to move text longer than 253 characters, you use the PROCESS GUI statement actions named EDIT-\*

### Attributes for Edit Area Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
MODIFIABLE	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TYPE	X	X/-	X
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

## Events

Change event (may be suppressed), enter event (may be suppressed), leave event (may be suppressed).

## Font Control

### Description

A font control is used to display the STRING attribute value of a dialog element in a certain font face, size and style. It is generated automatically if you select a font in the attributes window of a dialog element. You should, however, not rely on the automatic generation of specific font control names. You can also create a font control dynamically by using the NGU-FONT-SELECT dialog from library SYSTEM. When you assign the handle value of this font control to the FONT-HANDLE attribute of another dialog element, the STRING of the other dialog element is displayed accordingly.

#### Example:

```
#TC-1.FONT-HANDLE := #FNT-1
/* The STRING of the text-constant control
/* #TC-1 will be displayed in the face,
/* style and size of #FNT-1.
```

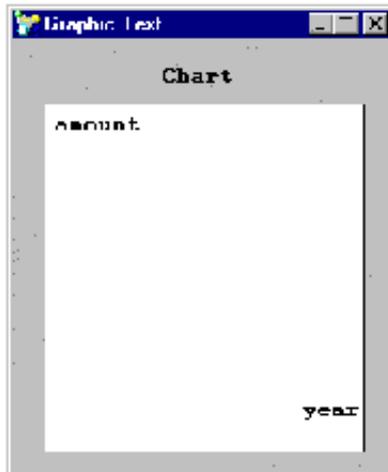
### Attributes for Font Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STRING	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	

### Events

This dialog element does not create events.

## Graphic-Text Control



### Description

A graphic-text control represents a one-line piece of text to be created on top of a canvas control. If it goes beyond the area of the canvas control, it is clipped.

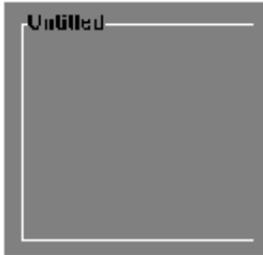
### Attributes for Graphic-Text Control

Attribute Name	Query	Set/Modify
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FONT-HANDLE	X	X/X
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PARENT	X	X/-
PREDECESSOR	X	-/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
STRING	X	X/X
SUCCESSOR	X	X/-
STYLE	X	-/-
TYPE	X	X/-
VISIBLE	X	X/X

## Events

This dialog element does not create events.

## Group-Frame Control



### Description

A group-frame control is used to optically group related dialog elements within a dialog. It can, for example, be used to frame a group of radio-button controls that have a common GROUP-ID, but where it is not optically evident to the end user that these radio-button controls are related.

You may use the group-frame control without the text in the upper left corner. Then the group-frame control acts as a simple frame. If you use it with text, this text may contain a mnemonic key (&). If the mnemonic key is pressed, the following dialog element in the navigation sequence gets the focus.

### Attributes for Group-Frame Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

This dialog element does not create events.

# GUI Control

## Description

In event handler code, you can use the HANDLE OF GUI variable to refer to the handle of any type of dialog element. This can be useful, for example, if you are querying an attribute value in all dialog elements on one level: you go through the dialog elements one after the other; in the course of this query, it is not clear which type of dialog element is going to be queried next. Then a GUI handle allows you to query the next dialog element regardless of its type. This saves a lot of coding, because otherwise, you would have to query each dialog element's attribute value separately.

The syntax checker will accept all existing attributes for a HANDLE OF GUI variable because all existing dialog elements must be covered. Nevertheless, if you query or modify an attribute which is not valid for the dialog element to which the HANDLE OF GUI variable refers, a runtime error will occur. It is therefore only advisable to use a HANDLE OF GUI variable if the attribute you are querying or modifying applies to most, if not all of the dialog elements.

### Example:

```
... 1 #CONTROL
HANDLE OF GUI
...
#CONTROL := #DLG$WINDOW.FIRST-CHILD REPEAT UNTIL #CONTROL = NULL-HANDLE
...
#CONTROL := #CONTROL.SUCCESSOR
END-REPEAT
```

**Attributes for GUI Control**

Attribute Name	Query	Set/Modify
ACCELERATOR	X	X/X
ACTIVE-CHILD	X	-/X
AUTOSELECT	X	X/X
BACKGROUND-COLOUR-NAME	X	X/X
BACKGROUND-COLOUR-VALUE	X	X/X
BAR-ID	X	X/-
BITMAP-FILE-NAME	X	X/X
CELL-ATTRIBUTES	X	X/X
CHECKED	X	X/X
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
COLUMN	X	X/X
COLUMN-COUNT	X	-/-
COLUMN-TYPE	X	X/-
CONTEXT-MENU	X	X/X
DEFAULT-BUTTON	X	X/X
DIL-TEXT	X	X/X
DOCKING	X	X/X
DRAGGABLE	X	X/X
EDIT-MASK	X	X/X
EMBEDDED-OBJECT	X	X/X
ENABLED	X	X/X
FIRST-CHILD	X	-/-
FIRST-COLUMN-WIDTH	X	X/X
FIRST-VISIBLE-COLUMN	X	X/X
FIRST-VISIBLE-ROW	X	X/X
FOLLOWS	X	X/X
FONT-HANDLE	X	X/X
FONT-STRING	X	X/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
FROZEN-COLUMNS	X	X/X

Attribute Name	Query	Set/Modify
GROUP-ID	X	X/X
HAS-DIL	X	X/X
HAS-FIRST-COLUMN	X	X/X
HAS-STATUS-BAR	X	X/X
HAS-SYSTEM-BUTTON	X	X/-
HAS-TOOLBAR	X	X/X
HAS-TOOLTIP	X	X/X
HEADER-FONT-HANDLE	X	X/X
HEADER-HEIGHT	X	X/X
HELP-ID	X	X/X
HORIZ-SCROLLABLE	X	X/-
ICONIZED	X	X/X
ITEM-H	X	X/X
ITEM-W	X	X/X
LAST-CHILD	X	-/-
LENGTH	X	X/X
LINE	X	X/X
LINKED	X	X/-
LOCATION	X	X/X
MARGIN-X	X	X/X
MARGIN-Y	X	X/X
MAX	X	X/X
MAXIMIZABLE	X	X/-
MAXIMIZED	X	X/X
MENU-HANDLE	X	X/X
MENU-ITEM-OLE	X	X/X
MENU-ITEM-TYPE	X	X/-
MIN	X	X/X
MINIMIZABLE	X	X/-
MINIMIZED	X	X/X
MODIFIABLE	X	X/-
MODIFIED	X	-/X
MODIFIED-SUCCESSOR	X	-/-
MULTI-SELECTION	X	X/-
NAME	X	-/-

Attribute Name	Query	Set/Modify
OBJECT-SIZE	X	X/X
OFFSET-X	X	X/X
OFFSET-Y	X	X/X
P1-X	X	X/X
P1-Y	X	X/X
P2-X	X	X/X
P2-Y	X	X/X
PAGE	X	X/X
PARENT	X	X/-
PREDECESSOR	X	-/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
ROW	X	X/X
ROW-COUNT	X	X/-
ROW-HEIGHT	X	X/X
SAME-AS	X	X/-
SCROLLRANGE-X	X	X/X
SCROLLRANGE-Y	X	X/X
SELECTED	X	X/X
SELECTED-SUCCESSOR	X	-/-
SERVER-OBJECT	X	X/X
SERVER-PROGID	X	X/X
SHARED	X	X/X
SIZE-MODIFIABLE	X	X/-
SLIDER	X	X/X
SORTED	X	X/-
STATUS-HANDLE	X	X/X
STATUS-TEXT	-	-/X
STRING	X	X/X
STYLE	X	X/-
SUCCESSOR	X	X/-
SUPPRESS-ACTIVATE-EVENT	X	X/X
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X

Attribute Name	Query	Set/Modify
SUPPRESS-CHANGE-EVENT	X	-/-
SUPPRESS-CLICK-EVENT	X	X/X
SUPPRESS-CLOSE-EVENT	X	X/X
SUPPRESS-COMMAND-STATUS-EVENT	X	X/X
SUPPRESS-DBL-CLICK-EVENT	X	X/X
SUPPRESS-DELETE-ROW-EVENT	X	X/X
SUPPRESS-ENTER-EVENT	X	X/X
SUPPRESS-ENTER-CELL-EVENT	X	X/X
SUPPRESS-FILL-EVENT	X	X/X
SUPPRESS-IDLE-EVENT	X	X/X
SUPPRESS-INSERT-ROW-EVENT	X	X/X
SUPPRESS-LEAVE-EVENT	X	X/X
SUPPRESS-LEAVE-CELL-EVENT	X	X/X
SUPPRESS-SIZE-EVENT	X	X/X
SUPPRESS-TOP-EVENT	X	X/X
TIMER-INTERVAL	X	X/X
TOOLBAR-HANDLE	X	X/X
TOOLBAR-POS	X	X/-
TOOLTIP	X	X/X
TYPE	X	X/-
VARIABLE	-	X/X
VERSION	X	-/-
VERT-SCROLLABLE	X	X/-
VISIBLE	X	X/X
ZOOM-FACTOR	X	X/X

## Input-Field Control



### Description

An input-field control is used to enter data in a single line. It corresponds to a field in a Natural program's INPUT statement. You can map database fields or other program variables to an input-field control by means of a linked variable. The input is automatically copied to a linked variable if you select the "Linked Variable" option in the attribute window's "Source" dialog box and you enter the name of the linked variable, for example, the database field. When the end user has finished entering data and the input-field control loses the focus, the data entered are validated.

You can also validate the input data by assigning an EDIT-MASK attribute to a (linked variable) input-field control: the check is performed when the input-field control loses the focus.

When the linked variables have been modified by code and you want to display the new values, you use the PROCESS GUI statement action REFRESH-LINKS.

Whenever input is rejected, for example, because the linked variable has another Natural data type, or because the EDIT-MASK was not matched, a message box is displayed that prompts the end user to "Retry" or "Cancel". The end user must press "Retry" to keep the current content of the input-field control and to modify it. The end user must press "Cancel" to reset the input-field control to the last valid content.

#### Notes:

When you create an input-field control, and you assign it a STYLE value of "Center" ('c') or "Right" ('r'), the input-field control's height must be greater than the height of the system font. Otherwise, the STRING will not be displayed.

Input in an input-field control is limited to 253 characters.

### Attributes for Input-Field Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
EDIT-MASK	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
LINKED	X	X/-	X
MODIFIABLE	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TYPE	X	X/-	
VARIABLE	-	X/X	X
VISIBLE	X	X/X	X

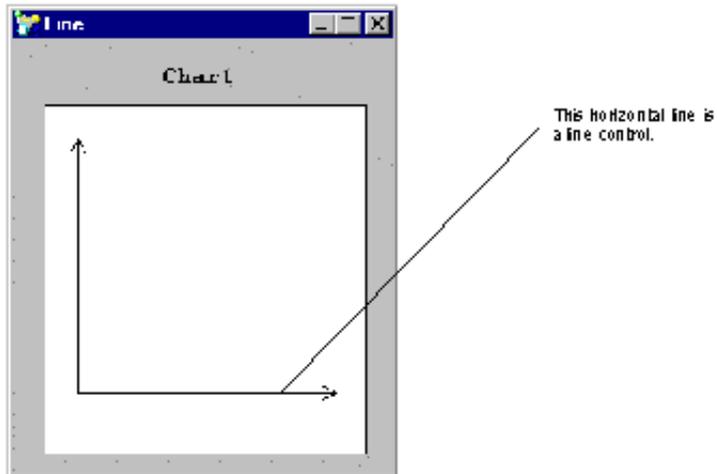
## Events

Change event (may be suppressed), enter event (may be suppressed), leave event (may be suppressed).

### Note:

The change event occurs for an input-field control if either the end user or the code changes the content. It does not occur on the initial setting of the input-field control. It is not recommended to manipulate the system focus from within the change event handler. Furthermore, the content of a linked variable is not updated in a change event.

## Line Control



### Description

A line control represents a line to be created in a canvas control. If it goes beyond the area of the canvas control, it is clipped.

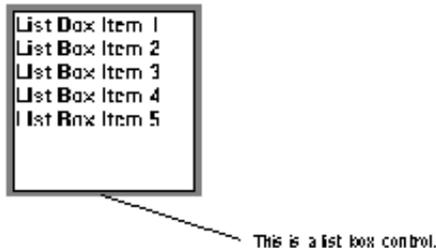
### Attributes for Line Control

Attribute Name	Query	Set/Modify
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PARENT	X	X/-
PREDECESSOR	X	-/-
P1-X	X	X/X
P1-Y	X	X/X
P2-X	X	X/X
P2-Y	X	X/X
STYLE	X	-/-
SUCCESSOR	X	X/-
TYPE	X	X/-
VISIBLE	X	X/X

## **Events**

This dialog element does not create events.

## List-Box Control



### Description

A list-box control contains a number of list-box items from which the end user can select one or more. You can insert items into a list-box control by:

- defining them in the dialog editor at design time, and by
- creating them dynamically at runtime using the PROCESS GUI statement actions ADD, ADD-ITEMS and ADD-ITEMS-EX.

The fill event enables you to implement dynamically growing list-box controls.

The MULTI-SELECTION attribute specifies for a list-box control whether the end user may select one or several items at a time.

#### Note:

A list-box control must have MULTI-SELECTION = TRUE to be able to use SELECTED-SUCCESSOR with the children list-box items.

To sort list-box item STRINGS alphabetically, you use the SORTED attribute for the list-box control. When the STRING of a list-box item is changed, the items are automatically sorted again. You can then go through the list of items with the SUCCESSOR attribute and the new sort sequence is reflected.

When a click event or a double-click event occurs for a list-box item, the list-box control receives the event and the list-box control's event handler can determine the clicked item by querying the attribute SELECTED-SUCCESSOR for the list-box control. When several list-box items are selected at the same time, the second selected item can be determined by querying the attribute SELECTED-SUCCESSOR of the first item, and so on.

If you specify event handlers for the click and the double-click events, the click-event handler will be executed prior to the double-click-event handler.

This may imply that the click-event handler suppresses the execution of the double-click-event handler (SUPPRESS-DBL-CLICK-EVENT attribute).

### Attributes for List-Box Control

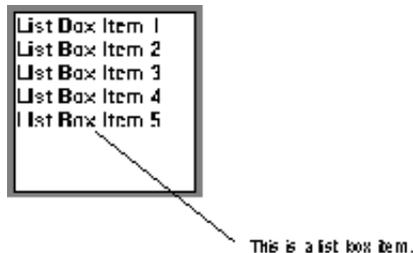
Attribute Name	Query	Set/Modify	In Attr. Window
AUTOSELECT	X	X/X	
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
MULTI-SELECTION	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	X
STYLE	X	-/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event (may be suppressed), double-click event (may be suppressed), fill event (may be suppressed).

## List-Box Item



### Description

A list-box item is an item inside a list-box control.

The list-box items are added to the list-box control either by specifying them in the dialog editor's List Box Attributes window, or by using the PROCESS GUI statement actions ADD-ITEMS, ADD-ITEMS-EX or ADD.

Items created using the dialog editor are represented in a Natural variable. The variable name is generated as: *list-box-handle-name-ITEMS (1:number-of-list-box-items)*. Example: #LB-1-ITEMS (1:5). The fifth item would then be called: #LB-1-ITEMS (5).

Items created with the PROCESS GUI statement can be given any variable name in the HANDLE definitions. Example: #MYITEM-1 HANDLE OF LISTBOXITEM.

To sort list-box item STRINGS alphabetically, you use the SORTED attribute for the list-box control. When the STRING of a list-box item is changed, the items are automatically sorted again. You can then go through the list of items with the SUCCESSOR attribute and the new sort sequence is reflected. For more information on how to go through a list of items, see **Working with List Box Controls and Selection-Box Controls** in the section Event-driven Programming Techniques of your Natural User's Guide.

A list-box item does not receive any events. Instead, the list-box control receives them, triggering event handlers there.

#### Note:

To select or deselect individual list-box items at runtime, hold down CTRL while clicking with the left mouse button.

### Attributes for List-Box Item

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SELECTED	X	X/X	
SELECTED-SUCCESSOR	X	-/-	
STRING	X	X/X	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

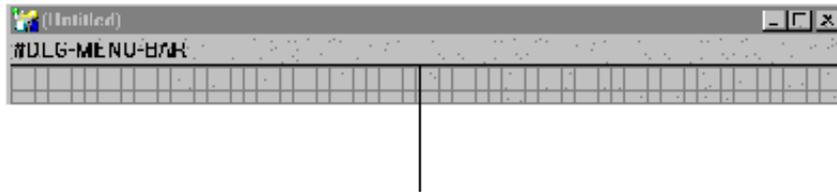
**Note:**

To be able to use SELECTED-SUCCESSOR, the parent list-box control must have MULTI-SELECTION = TRUE.

**Events**

This dialog element does not create events.

## Menu Bar



This is a menu bar control without any menu items or submenu controls.

### Description

A menu bar is displayed at the top of the dialog window. It is the top level of a menu structure and contains menu items (second level). A menu item may be of type submenu. Then it is a submenu control which may be pulled down; it contains menu items (third level). The number of levels is unlimited.

A menu bar only becomes visible in a dialog when the attribute MENU-HANDLE is set to a value for the dialog. The PARENT of a menu bar may be a dialog or a NULL-HANDLE. If a dialog is the PARENT, the menu structure is specific to the dialog and is deleted when the dialog is deleted. If a NULL-HANDLE is the PARENT, the menu structure is free and will be closed when the application is closed. A free menu structure can be shared among several dialogs when the handles are defined in a global data area rather than in a local data area.

For an MDI child window, the menu bar is displayed only at the top of the MDI frame window, not in the MDI child window. The MDI children all share one menu bar, which is displayed in the MDI frame window. Every time another MDI child window is activated, the menu bar changes to reflect the menu bar defined for the particular MDI child window.

#### Note:

You may create only one menu bar per dialog. By default, the dialog editor generates a menu bar named "#DLG-MENU-BAR", the parent being the dialog window.

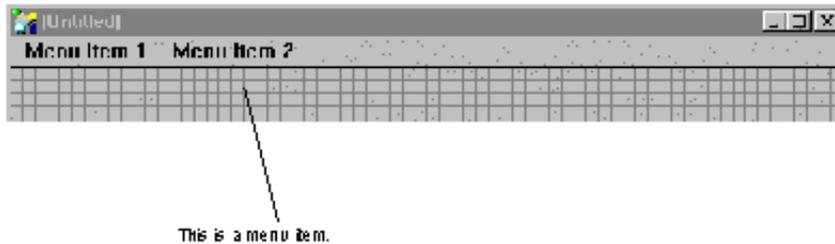
### Attributes for Menu Bar

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	

## **Events**

This dialog element does not create events.

## Menu Item



### Description

A menu item is an item inside a menu bar or a submenu control. It is a child of a menu bar or a submenu control.

There are several types of menu items (values of the attribute `MENU-ITEM-TYPE`). Menu items of type `MT-NORMAL` trigger the click-event handler when an end user clicks on them. The `MT-SUBMENU` type of menu item is associated with a submenu control, which is pulled down when an end user clicks on the menu item.

It is not recommended to define menu items of type `MT-NORMAL` on the first level of a menu bar. Instead, you should define menu items of type `MT-SUBMENU`. When you give a name to a menu item by setting the `STRING` attribute to a name used by the windowing system, such as "Cascade", you must not modify this `STRING` value dynamically.

### Attributes for Menu Item

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR <sup>3)</sup>	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED <sup>1)</sup>	X	X/X	X
CLIENT-DATA <sup>2)</sup>	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT <sup>2)</sup>	X	X/X	X
ENABLED <sup>3)</sup>	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-HANDLE <sup>4)</sup>	X	X/-	
MENU-ITEM-OLE	X	X/X	
MENU-ITEM-TYPE <sup>5)</sup>	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SAME-AS	X	X/X	X
SHARED	X	X/X	X
STRING <sup>3)</sup>	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TOOLTIP	X	-/-	
TYPE	X	X/-	

1) For MENU-ITEM-TYPE = MT-NORMAL only. Not for menu items inside the menu bar.

2) For MENU-ITEM-TYPEs MT-NORMAL and MT-MDI only.

3) Not for MENU-ITEM-TYPE = MT-SEPARATOR.

4) For MENU-ITEM-TYPEs MT-SUBMENU and MT-WINDOWMENU only.

5) If MENU-ITEM-TYPE = MT-WINDOWMENU, the menu item must be inside the menu bar (at the top level).

## Events

Click event.

## OLE Container Control

17 Reg on A
25 Reg on D
77 Reg on C
118 Reg on Total

This is an OLE container control with an embedded spreadsheet.

### Description

An OLE container control enables you to integrate OLE objects in a Natural dialog.

### Attributes for OLE Container Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
EMBEDDED-OBJECT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
ICONIZED	X	X/-	
INPLACE-ACTIVE	X	-/-	
LAST-CHILD	X	-/-	
MODIFIABLE	X	X/X	X
OBJECT-SIZE	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SERVER-OBJECT	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
SERVER-PROGID	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-ACTIVATE-EVENT	X	X/X	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-CLOSE-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X
ZOOM-FACTOR	X	X/X	X

## Events

Activate event (may be suppressed), change event (may be suppressed), click event (may be suppressed), close event (may be suppressed), double-click event (may be suppressed).

## Push-Button Control



### Description

A push-button control is a representation of a button. The end user can click on it to trigger a certain action (specified in the click-event handler code). You can use a push-button control's handle value in the DEFAULT-BUTTON attribute of a dialog; your push-button control then serves as the dialog's default button.

### Attributes for Push-Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event.

## Radio-Button Control



These three radio button controls have been created independently; they share a GROUP-ID, and one is checked.

### Description

A radio-button control is a selection item. You can logically group several radio-button controls by assigning them a common GROUP-ID attribute. Out of this group, none or one may be selected by default. To pre-select a radio-button control, you assign it a CHECKED attribute with a value of CHECKED. Please note that only the radio-button control CHECKED last will remain CHECKED; the one CHECKED before will be unchecked automatically.

You use it, for example, to let the end user switch an option on or off, affecting the status of other radio-button controls of the same group. A change in the state of a toggle-button control, by contrast, will not affect the state of other toggle-button controls in the same dialog.

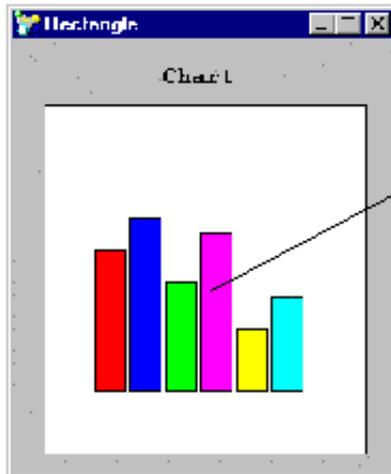
### Attributes for Radio-Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
GROUP-ID	X	X/-	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## **Events**

Click event (may be suppressed).

## Rectangle Control



This is a rectangle control.

### Description

A rectangle control represents a filled rectangle to be created in of a canvas control. If it goes beyond the area of the canvas control, it is clipped.

### Attributes for Rectangle Control

Attribute Name	Query	Set/Modify
BACKGROUND-COLOUR-NAME	X	X/X
BACKGROUND-COLOUR-VALUE	X	X/X
CLIENT-DATA	X	X/X
CLIENT-HANDLE	X	X/X
CLIENT-KEY	X	X/X
CLIENT-VALUE	X	X/X
FIRST-CHILD	X	-/-
FOREGROUND-COLOUR-NAME	X	X/X
FOREGROUND-COLOUR-VALUE	X	X/X
LAST-CHILD	X	-/-
PREDECESSOR	X	-/-
PARENT	X	X/-
RECTANGLE-H	X	X/X
RECTANGLE-W	X	X/X
RECTANGLE-X	X	X/X
RECTANGLE-Y	X	X/X
STYLE	X	X/-
SUCCESSOR	X	X/-
TYPE	X	X/-
VISIBLE	X	X/X

## Events

This dialog element does not create events.

## Scrollbar Control



### Description

A scroll bar control enables the end user to select a position on a scale. If, for example, you want the end user to select a percentage value on a scale from 1 to 100, you can use a scroll bar control and tell the end user in the **STRING** attribute of a text-constant control which percentage value (integer) was selected. To limit the scale, you must always determine values for the **MIN** and **MAX** attributes of your scroll bar control. You can then use the **LINE** and **PAGE** attributes to determine step sizes for scrolling with the arrow buttons (**LINE**) or the scroll bar shaft (**PAGE**). You can also set and query the position of the slider with the **SLIDER** attribute.

#### Note:

The range of the dialog scroll bar control attributes **min**, **max**, **page**, **line** and **slider** has been increased. Now for all of these attributes a range of -1073741823 to 1073741823 is permitted.

### Attributes for Scrollbar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LINE	X	X/X	X
MAX	X	X/X	X
MIN	X	X/X	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PAGE	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SLIDER	X	X/X	X
STYLE	-	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Change event (may be suppressed).

## Selection-Box Control



This is a selection box control in its original state; when you click on the arrow push button, a list box drops down.

### Description

A selection-box control is a combination of an input-field, a list-box, and a push button. When no selection has been made, it consists of an input-field with a push button next to it. With the push button, the end user may open a list-box from the input-field. This list-box contains any number of items from which the end user can select one. The selected item is then copied into the input-field.

It is possible to get a selection-box control without a push button where the list-box is always dropped down: the STYLE attribute value must have the value "X".

You can map database fields or other program variables to the input section of a selection-box control by means of a linked variable. The input is automatically copied to a linked variable if you select the "Linked Variable" option in the attribute window's "Source" dialog box and you enter the name of the linked variable, for example the database field. When the end user has finished entering data and the selection-box control loses the focus, the entered data are validated.

You can also validate the input data by assigning an EDIT-MASK attribute to the selection-box control: the check is performed when the input-field in the selection-box control loses the focus.

When the linked variables have been modified by code and you want to display the new values, you use the PROCESS GUI statement action REFRESH-LINKS.

Whenever input is rejected, for example, because the linked variable has another Natural data type, or because the EDIT-MASK was not matched, a message box is displayed that prompts the end user to "Retry" or "Cancel". The end user must press "Retry" to keep the current content of the input-field control and to modify it. The end user must press "Cancel" to reset the input-field control to the last valid content.

#### Note:

Input in a selection-box control's input-field is limited to 253 characters.

If you set the STYLE attribute to the value "X" (dropped down), the MODIFIABLE attribute value is automatically set to TRUE. If you set the MODIFIABLE attribute to FALSE, the STYLE attribute automatically no longer has the value "X".

### Attributes for Selection-Box Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
EDIT-MASK	X	X/-	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
LENGTH	X	X/X	X
LINKED	X	X/-	X
MODIFIABLE	X	X/-	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
SELECTED-SUCCESSOR	X	-/-	
SORTED	X	X/-	
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-LEAVE-EVENT	X	X/X	
TYPE	X	X/-	
VARIABLE		X/X	X
VISIBLE	X	X/X	X

## Events

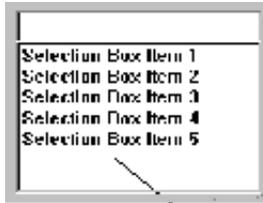
Change event (may be suppressed), enter event (may be suppressed), leave event (may be suppressed).

**Note:**

The change event occurs for a selection-box control if the end user changes the content, if the code changes the content or if an item is selected and transferred to the input-field area. It does not occur on the initial setting of the selection-box control.

It is not recommended to manipulate the system focus from within the change event handler.

## Selection-Box Item



This is a selection box item inside a selection box control where the list box has been pulled down.

### Description

A selection-box item is an item inside a selection-box control.

The selection-box items are added to the list-box control either by specifying them in the dialog editor's selection-box attributes window, or by using the PROCESS GUI statement actions ADD-ITEMS, ADD-ITEMS-EX or ADD. You can find out the selected item by querying the STRING attribute of the selection-box control.

Items created using the dialog editor are represented in a Natural variable. The variable name is generated as: *selection-box-handle-name-ITEMS (1:number-of-selection-box-items)*.

Example: #SB-1-ITEMS (1:5). The fifth item would then be called: #SB-1-ITEMS (5).

Items created with the PROCESS GUI statement actions can be given any variable name in the HANDLE definitions.

Example: #MYITEM-1 HANDLE OF SELECTIONBOXITEM.

A selection-box item does not receive any events. Instead, the selection-box control receives them, triggering event handlers there.

### Attributes for Selection-Box Item

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
FIRST-CHILD	X	-/-	
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STRING	X	X/X	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	

## Events

This dialog element does not create events.

# Signal

## Description

A signal is an abstract dialog element, which does not have a user interface of its own. It is used to represent an application action which can be triggered via any number of menu items and/or tool-bar items, by linking these items to the signal via their SAME-AS attribute.

By representing each program action by a signal, and linking to the signal from each menu or tool-bar item which triggers this action, the relevant attribute values only need to be specified once for the signal itself. The signal's attributes are automatically inherited by all items which are linked to it. This inheritance mechanism also applies to any modifications made to the signal after it has been created. For example, if the ENABLED attribute of the signal is set to FALSE, all menu items and tool-bar items linked to it will be automatically disabled

If a menu or tool-bar item that is linked to a signal is clicked, a Click Event is raised for the signal itself, rather than for the menu or tool-bar item, ensuring that the same code is invoked for all items which are linked to the same signal.

## Attributes for Signal

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-ITEM-TYPE	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SHARED	X	X/X	X
SUCCESSOR	X	-/-	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

## **Events**

Click event.

## Status-Bar Control



### Description

A status-bar control is an alternative to the traditional status-bar that is created by setting the dialog's HAS-STATUS-BAR attribute. However, the status-bar control offers a range of advanced features which the traditional Natural status-bars do not support. These new features include a Windows-like appearance with (optionally) recessed or raised sections - referred to as "panes" - and (optionally) a sizing grip (see illustration above). The status-bar panes are themselves dialog elements, and are therefore documented under their own section below. Note that, although it is possible to define a status-bar control with no panes, such a status-bar control can only display a single text string. In order to use any of the advanced features, it is necessary to define one or more panes.

A status-bar control can be displayed either at the top or bottom of a dialog. Furthermore, a dialog may have more than one status-bar control. Status information set via the dialog's STATUS-TEXT attribute is automatically redirected to the status-bar control (if any) specified by the dialog's STATUS-HANDLE attribute.

You can set a minimum pane height for the status-bar control using the ITEM-H attribute. The distance between the items and the status-bar control's border is determined with the MARGIN-X and MARGIN-Y attributes.

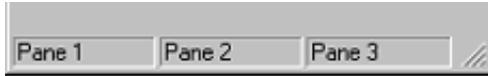
### Attributes for Status-Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BAR-ID	X	X/-	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
Context-MENU	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
HAS-TOOLTIP	X	X/X	X
ITEM-H	X	X/X	X
LAST-CHILD	X	-/-	
LOCATION	X	X/X	X
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	-/-	
RECTANGLE-W	X	-/-	
RECTANGLE-X	X	-/-	
RECTANGLE-Y	X	-/-	
STRING	X	-/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

This dialog element does not create events.

## Status-Bar Pane



### Description

A status-bar pane is a logical section of a status-bar control. You can define more panes than are physically visible in the control by selectively setting the **VISIBLE** attribute of those logical panes which should not appear, according to the application context (for example, depending on which MDI child dialog is active).

A pane can be stretchy or non-stretchy. A stretchy pane is indicated by setting the pane's **RECTANGLE-W** attribute to zero. The width of a stretchy pane changes when the width of the parent dialog is changed, and is determined by the following formula:

**width = (available width - total width of all non-stretchy panes) / no. of stretchy panes**

The first (visible) stretchy pane is referred to as the message pane, and is used as the default pane in cases where no pane is explicitly specified. This is the case, for example, when text is output by setting the status-bar control's **STRING** attribute or the dialog's **STATUS-TEXT** attribute.

Both an icon and a string can be specified for a pane by setting the pane's **BITMAP-FILE-NAME** and **STRING** attributes, respectively. If the **ENABLED** attribute is set to **FALSE**, both the icon and string are either grayed out or completely suppressed (depending on a pane-style flag setting). Furthermore, the status-bar pane supports the **TOOLTIP** attributes, allowing pane-specific tool tips to be specified when the mouse pointer hovers over the pane.

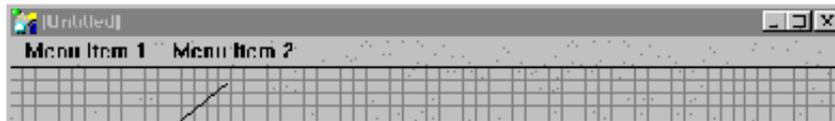
### Attributes for Status-Bar Pane

Attribute Name	Query	Set/Modify	In Attr. Window
BITMAP-FILE-NAME	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-W	X	X/X	X
SHARED	X	X/X	X
STRING	X	-/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event (may be suppressed), double-click event (may be suppressed).

## Submenu Control



This menu item represents a submenu control; that is, when you click on the menu item, a submenu will be pulled down.

### Description

A submenu control is a part of a complex menu structure. When the end user chooses a menu item of type submenu, the submenu control is pulled down, containing a vertical list of menu items.

You create a submenu control in the dialog editor by creating a menu item with the MENU-ITEM-TYPE of MT-SUBMENU or MT-WINDOWMENU. In Natural code, a submenu control is embedded into a menu structure by assigning its HANDLE to the MENU-HANDLE attribute of a menu item.

An MT-WINDOWMENU submenu control is used for MDI frame and MDI child windows. The windowing system adds the currently existing MDI child windows dynamically to such a submenu control.

The PARENT of a submenu control may be a dialog or NULL-HANDLE. If a dialog is the PARENT, the submenu control is private to the dialog and is deleted when the dialog is closed. If a NULL-HANDLE is the PARENT, the submenu control is free and will be deleted when the application is closed. A free submenu control can be shared among several menu structures in several dialogs if the handles are defined in a global data area rather than in a local data area.

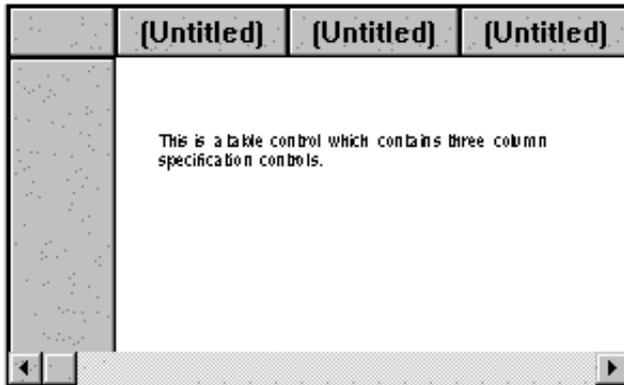
### Attributes for Submenu Control

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
HELP-ID	X	X/X	
LAST-CHILD	X	-/-	
MENU-ITEM-OLE	X	X/X	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/-	
SUCCESSOR	X	-/-	
SUPPRESS-BEFORE-OPEN-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	

## Events

Before-open event (may be suppressed).

## Table Control



### Description

A table control is a dialog element representing a spreadsheet. It provides an optional header row for naming the columns and an optional left column for naming the rows.

Once the table control has defined the spreadsheet as such, the columns are defined by adding column-specification controls. The cells of the table control are not defined as dialog elements but are identified by the values of the table control's COLUMN and ROW attributes. The first cell is identified by ROW=1 and COLUMN=1. The cells of each column may be used differently depending on the value of the column-specification control's COLUMN-TYPE attribute: they may be used as input-fields, selection-boxes with items or toggle-buttons.

You manipulate table controls with a set of PROCESS GUI statement actions.

### Attributes for Table Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CELL-ATTRIBUTES	X	X/X	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
COLUMN	X	-/X	
COLUMN-COUNT	X	X/-	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FIRST-COLUMN-WIDTH	X	X/X	X

Attribute Name	Query	Set/Modify	In Attr. Window
FIRST-VISIBLE-COLUMN	X	X/X	X
FIRST-VISIBLE-ROW	X	X/X	X
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
FROZEN-COLUMNS	X	X/X	X
HAS-FIRST-COLUMN	X	X/-	X
HEADER-FONT-HANDLE	X	X/X	X
HEADER-HEIGHT	X	X/X	X
HELP-ID	X	X/X	X
HORIZ-SCROLLABLE	X	X/-	X
LAST-CHILD	X	-/-	
MODIFIABLE	X	X/-	X
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
ROW	X	-/X	
ROW-COUNT	X	X/-	X
ROW-HEIGHT	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
SUPPRESS-CHANGE-EVENT	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
SUPPRESS-DBL-CLICK-EVENT	X	X/X	
SUPPRESS-DELETE-ROW-EVENT	X	X/X	
SUPPRESS-ENTER-EVENT	X	X/X	
SUPPRESS-ENTER-CELL-EVENT	X	X/X	
SUPPRESS-FILL-EVENT	X	X/X	
SUPPRESS-INSERT-ROW-EVENT	X	X/X	

Attribute Name	Query	Set/Modify	In Attr. Window
SUPPRESS-LEAVE-EVENT	X	X/X	
SUPPRESS-LEAVE-CELL-EVENT	X	X/X	
SUPPRESS-TOP-EVENT	X	X/X	
TYPE	X	X/-	
VERT-SCROLLABLE	X	X/-	X
VISIBLE	X	X/X	X

### Attributes for Cells in a Table Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	
BACKGROUND-COLOUR-VALUE	X	X/X	
CHECKED	X	X/X	
COLUMN	X	X/X	
DIL-TEXT	X	X/X	
FOREGROUND-COLOUR-NAME	X	X/X	
FOREGROUND-COLOUR-VALUE	X	X/X	
ROW	X	X/X	
STRING	X	X/X	

### COLUMN-TYPEs in Cells and Their Attributes

The attributes that are valid for cells in table controls in general are not always available depending on the value of the column-specification control's attribute COLUMN-TYPE. Such a value might be, for example, "input-field control". The cell can then be used like an input-field control. For this input-field cell, only a subset of the attributes is available.

The following table specifies which attributes are *not* applicable to specific COLUMN-TYPEs.

COLUMN-TYPE	Attributes NOT Available
Input field control	CHECKED
Selection box control	CHECKED
Toggle button control	STRING

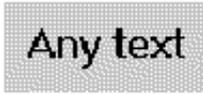
### Events

Change event, click event, delete-row event, dbl-click-event, enter event, enter-cell event, fill event, insert-row event, leave event, leave-cell event, top event. All events may be suppressed.

The following events are not always triggered:

- click event - only for toggle-button fields,
- dbl-click event - for input-fields and toggle-buttons,
- change event - only for input-fields and for selection-boxes.

## Text-Constant Control



### Description

A text-constant control displays non-modifiable text within a dialog. You use it, for example, to describe which data can be entered in an adjacent input-field control. The text STRING in text-constant controls may also contain a mnemonic key (&). If the end user presses the mnemonic key, the next input dialog element in the navigation sequence gets the focus.

### Attributes for Text-Constant Control

Attribute Name	Query	Set/Modify	In Attr. Window
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## **Events**

This dialog element does not create events.

# Timer

## Description

A timer is a dialog element which is invisible to the end user but which allows events to be triggered periodically in the dialog. You can, for example, update the STRING attribute of a text-constant control with the click event of a timer.

### Example:

```
#TC-1.STRING:= *TIMX
/* Display the system time in the text-constant
/* control #TC-1 and update the time (= timer /* control click event) every 1000 milliseconds
/* (TIMER-INTERVAL attribute).
```

### Note:

It is not recommended to write code that counts clicks in the timer click-event handler, because the click events that occur while the system is busy may get lost.

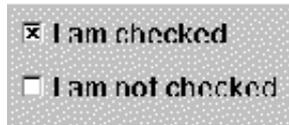
## Attributes for Timer

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TIMER-INTERVAL	X	X/X	
TYPE	X	X/-	

## Events

Click event (may be suppressed).

## Toggle-Button Control



### Description

A toggle-button control offers the end user a checked/not checked (on/off) alternative. You use it, for example, to let the end user switch an option on or off without affecting the status of other adjacent toggle-button controls. A change in the state of a radio-button control, by contrast, will affect the state of other radio-button controls that belong to the same group.

### Attributes for Toggle-Button Control

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
CONTEXT-MENU	X	X/X	X
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
FONT-HANDLE	X	X/X	X
FOREGROUND-COLOUR-NAME	X	X/X	X
FOREGROUND-COLOUR-VALUE	X	X/X	X
HELP-ID	X	X/X	X
LAST-CHILD	X	-/-	
MODIFIED	X	-/X	
MODIFIED-SUCCESSOR	X	-/-	
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/X	X
RECTANGLE-W	X	X/X	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
SUCCESSOR	X	-/-	
SUPPRESS-CLICK-EVENT	X	X/X	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event (may be suppressed).

## Tool Bar



This is a toolbar control without any toolbar items.

### Description

A tool bar is similar to a menu bar. It consists of horizontally or vertically grouped tool-bar items with a bitmap on each of them, providing quick access to the most frequently needed program functions. By default, it is located at the top of the dialog, directly below the menu bar. It can also be located at the bottom, the left or the right of the dialog window.

A tool bar only becomes visible within a dialog when its handle value is assigned to the dialog's TOOLBAR-HANDLE attribute and HAS-TOOLBAR is TRUE for the dialog. When you create the tool bar with the dialog editor, the dialog editor does this automatically. For MDI child windows, the tool bar is displayed at the top of the MDI frame window rather than in the child window itself. Any time another MDI child window is activated, the tool bar changes to reflect the tool bar defined for the particular MDI child window.

You can determine the size of the tool-bar items using the ITEM-W and ITEM-H attributes. The distance between the items and the tool bar's border is determined with the MARGIN-X and MARGIN-Y attributes. If you set the tool bar's style to "w", the items that would exceed the tool bar's width are wrapped around.

### Attributes for Tool Bar

Attribute Name	Query	Set/Modify	In Attr. Window
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	X
CLIENT-VALUE	X	X/X	
FIRST-CHILD	X	-/-	
ITEM-H	X	X/X	X
ITEM-W	X	X/X	X
LAST-CHILD	X	-/-	
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
STYLE	X	X/X	X
SUCCESSOR	X	-/-	
TYPE	X	X/-	

## **Events**

This dialog element does not create events.

## Tool-Bar Control



### Description

A tool-bar control is an alternative to the traditional tool bar described above. It, too, can contain one or more tool-bar items used to trigger specific application functionality. However, the tool-bar control offers a range of advanced features which the traditional Natural tool bars do not support. These new features include a Windows-like appearance with (optionally) flat buttons, the ability to specify that the tool bar should be "dockable" (see below), the ability to embed other dialog elements (such as selection-boxes), and the ability to define tool tips for the tool-bar items.

A tool-bar control is dockable if its DRAGGABLE attribute is set to TRUE when the control is created. Dockable tool bars have the advantage that they can be repositioned by the user by tearing off the tool-bar control via clicking on the gripper (see illustration above), a separator or any part of its background area and dragging it with the primary mouse button held down. When the tool-bar control is dropped, it will either be snapped into position at the side of the dialog ("docked"), or "floated" in its own window. Whether the control is floated or docked depends on where the control was dropped, whether the dialog and tool-bar control both allow docking on the target side and whether the CTRL key is being held down. In the latter case, the tool bar will be floated regardless of its drop position. The border width of the drag rectangle informs the user as to whether the tool-bar control will be floated (thick border) or docked (thin border). The dragging process can be aborted at any time by pressing the ESC key.

A further major benefit of using tool-bar controls instead of the traditional tool bars is that multiple tool-bar controls can be used within a dialog, which can be independently docked or floated as required. The layout can be preserved on a per-user basis between sessions by activating the "save layout" option in the Dialog Attributes window in the dialog editor.

You can determine the size of the tool-bar items using the ITEM-W and ITEM-H attributes. The distance between the items and the tool-bar control's border is determined with the MARGIN-X and MARGIN-Y attributes.

### Attributes for Tool-Bar Control

Attribute Name	Query	Set/Modify	In Attr. Window
BAR-ID	X	X/-	
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	X
CONTEXT-MENU	X	X/X	X
DOCKING	X	X/X	X
DRAGGABLE	X	X/-	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
FOLLOWS	X	X/X	
HAS-DIL	X	X/X	X
HAS-TOOLTIP	X	X/X	X
ITEM-H	X	X/-	X
ITEM-W	X	X/-	X
LAST-CHILD	X	-/-	
LOCATION	X	X/X	X
MARGIN-X	X	X/X	X
MARGIN-Y	X	X/X	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
RECTANGLE-H	X	X/-	X
RECTANGLE-W	X	X/-	X
RECTANGLE-X	X	X/X	X
RECTANGLE-Y	X	X/X	X
STRING	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

This dialog element does not create events.

## Tool-Bar Item



This is a toolbar item.

### Description

A tool-bar item is an item inside a tool bar. A tool-bar item is similar to a push button with a bitmap displayed on it. In most cases, a tool-bar item will serve as an alias for a menu item. So, instead of programming again the same click-event handler code as for the menu item, you can use the tool-bar item's `SAME-AS` attribute and assign the corresponding menu item handle value. If the tool-bar item is clicked, the event handler of the corresponding menu item is triggered. If you do not use the `SAME-AS` attribute, you must specify the event handler code with the tool-bar item.

You can determine the size of the tool-bar items using the parent tool bar's `ITEM-W` and `ITEM-H` attributes.

#### Note:

For performance reasons, you should not use more than 16 colors in tool bar icon bitmaps; all bitmaps should share the same palette of colors.

### Attributes for Tool-Bar Item

Attribute Name	Query	Set/Modify	In Attr. Window
ACCELERATOR	X	X/X	X
BACKGROUND-COLOUR-NAME	X	X/X	X
BACKGROUND-COLOUR-VALUE	X	X/X	X
BITMAP-FILE-NAME	X	X/X	X
CHECKED	X	X/X	X
CLIENT-DATA	X	X/X	
CLIENT-HANDLE	X	X/X	
CLIENT-KEY	X	X/X	
CLIENT-VALUE	X	X/X	
DIL-TEXT	X	X/X	X
ENABLED	X	X/X	X
FIRST-CHILD	X	-/-	
LAST-CHILD	X	-/-	
MENU-ITEM-TYPE	X	X/-	X
PARENT	X	X/-	
PREDECESSOR	X	-/-	
SAME-AS	X	X/X	X
SHARED	X	X/X	X
STYLE	X	X/-	X
SUCCESSOR	X	X/-	
TOOLTIP	X	X/X	X
TYPE	X	X/-	
VISIBLE	X	X/X	X

## Events

Click event.

# Attributes

## Alphabetical List of Attributes

- ACCELERATOR
- ACTIVE-CHILD
- AUTOADJUST
- AUTOSELECT
- BACKGROUND-COLOUR-NAME
- BACKGROUND-COLOUR-VALUE
- BAR-ID
- BITMAP-FILE-NAME
- CELL-ATTRIBUTES
- CHECKED
- CLIENT-DATA
- CLIENT-HANDLE
- CLIENT-KEY
- CLIENT-VALUE
- COLUMN
- COLUMN-COUNT
- COLUMN-TYPE
- CONTEXT-MENU
- DEFAULT-BUTTON
- DIL-TEXT
- DOCKING
- DPI
- DRAGGABLE
- EDIT-MASK
- EMBEDDED-OBJECT
- ENABLED
- EVENT-QUEUEING
- FIRST-CHILD
- FIRST-COLUMN-WIDTH
- FIRST-VISIBLE-COLUMN
- FIRST-VISIBLE-ROW
- FOLLOWS
- FONT-HANDLE
- FONT-STRING
- FOREGROUND-COLOUR-NAME
- FOREGROUND-COLOUR-VALUE
- FROZEN-COLUMNS
- GROUP-ID
- HANDLE-VARIABLE
- HAS-DIL
- HAS-FIRST-COLUMN
- HAS-HELP-BUTTON
- HAS-MENU-BAR
- HAS-STATUS-BAR
- HAS-SYSTEM-BUTTON
- HAS-TOOLBAR

- HAS-TOOLTIP
- HEADER-FONT-HANDLE
- HEADER-HEIGHT
- HELP-FILENAME
- HELP-ID
- HORIZ-SCROLLABLE
- ICONIZED
- INPLACE-ACTIVE
- ITEM-H
- ITEM-W
- LAST-CHILD
- LENGTH
- LINE
- LINKED
- LOCATION
- MARGIN-X
- MARGIN-Y
- MAX
- MAXIMIZABLE
- MAXIMIZED
- MENU-HANDLE
- MENU-ITEM-OLE
- MENU-ITEM-TYPE
- MIN
- MINIMIZABLE
- MINIMIZED
- MODIFIABLE
- MODIFIED
- MODIFIED-SUCCESSOR
- MULTI-SELECTION
- NAME
- OBJECT-SIZE
- OFFSET-X
- OFFSET-Y
- P1-X
- P2-X
- P2-Y
- PAGE
- PARENT
- POPUP-HELP
- PREDECESSOR
- RECTANGLE-H
- RECTANGLE-W
- RECTANGLE-X
- RECTANGLE-Y
- ROW
- ROW-COUNT
- ROW-HEIGHT
- SAME-AS
- SCROLLRANGE-X
- SCROLLRANGE-Y
- SELECTED

- SELECTED-SUCCESSOR
  - SERVER-OBJECT
  - SERVER-PROGID
  - SHARED
  - SIZE-MODIFIABLE
  - SLIDER
  - SORTED
  - STATUS-HANDLE
  - STATUS-TEXT
  - STRING
  - STYLE
  - SUCCESSOR
  - SUPPRESS-ACTIVATE-EVENT
  - SUPPRESS-BEFORE-OPEN-EVENT
  - SUPPRESS-CHANGE-EVENT
  - SUPPRESS-CLICK-EVENT
  - SUPPRESS-CLIENT-SIZE-EVENT
  - SUPPRESS-CLOSE-EVENT
  - SUPPRESS-COMMAND-STATUS-EVENT
  - SUPPRESS-DBL-CLICK-EVENT
  - SUPPRESS-DELETE-ROW-EVENT
  - SUPPRESS-ENTER-EVENT
  - SUPPRESS-ENTER-CELL-EVENT
  - SUPPRESS-FILL-EVENT
  - SUPPRESS-IDLE-EVENT
  - SUPPRESS-INSERT-ROW-EVENT
  - SUPPRESS-LEAVE-EVENT
  - SUPPRESS-LEAVE-CELL-EVENT
  - SUPPRESS-SIZE-EVENT
  - SUPPRESS-TOP-EVENT
  - TIMER-INTERVAL
  - TOOLBAR-HANDLE
  - TOOLBAR-POS
  - TYPE
  - VARIABLE
  - VERSION
  - VERT-SCROLLABLE
  - VISIBLE
  - ZOOM-FACTOR
- 

## ACCELERATOR

Enables you to define accelerator keys. If the end user presses an accelerator key, the double-click event occurs for the dialog element, or if no double-click event is available, the click event occurs. The accelerator key does not work if the corresponding event is suppressed.

**Note:**

User-defined accelerator keys overwrite identical operating system accelerator keys, with the exception of hot keys such as "Alt+Tab", "Ctrl+Esc", "Ctrl+Alt+Del", etc.

<b>Applies to</b>	Bitmap control, list box control, menu item, OLE container control, push button control, radio button control, signal, toggle button control, tool bar item.
<b>Data Type</b>	A253
<b>Default Value</b>	None
<b>Possible Values</b>	<p>The standard syntax is:  [Ctrl+][Alt+][Shift+]&lt;key&gt;  where &lt;key&gt; can be any of the following values:  "A" to "Z"  "0" to "9" (but not in conjunction with shift key)  "F1" to "F12" (but not F1 on its own)  "Num 0" to "Num 9" (=numeric keypad number keys)  "Num Dec" (=numeric keypad decimal point key)  "Num +" (=numeric keypad add key)  "Num -" (=numeric keypad subtract key)  "Num *" (=numeric keypad multiply key)  "Num /" (=numeric keypad divide key)  "Space"  "Backspace"  "Tab"  "Enter" (=return key)  "Esc" (=escape key)  "Ins" (=insert key)  "Del" (=delete key)  "Home"  "End"  "Page Up" or "Page Down"  "Left Arrow", "Right Arrow", "Up Arrow" or "Down Arrow"  Examples: "3", "F7", "Shift+Page Up", "Alt+Enter", "Ctrl+Alt+G". Note that the key modifiers ("Ctrl", "Alt" and "Shift") can be specified in any order. Accelerators are not case-sensitive. Please keep in mind that F1 cannot be an accelerator and therefore help topics cannot be invoked this way.</p>

## ACTIVE-CHILD

Specifies the active MDI child dialog (if any) for the specified MDI frame dialog.

<b>Applies to</b>	Dialog (MDI frame).
<b>Data Type</b>	HANDLE
<b>Default Value</b>	None
<b>Possible Values</b>	NULL-HANDLE / MDI child dialog handle

## AUTOADJUST

Indicates whether the coordinates specified on dialog element creation are to be scaled in the case of scalable dialogs (i.e., dialogs created with a non-zero DPI attribute). This is useful for dynamically creating controls in scalable dialogs whose size and/or position is based on an existing dialog element.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE for scalable dialogs; FALSE otherwise.
<b>Possible Values</b>	TRUE / FALSE

## AUTOSELECT

Specifies whether the list box item under the mouse cursor is to be selected automatically, if not already selected, before a context menu is displayed. If the list box item is not already selected, the existing selection (if any) is cleared before selecting it in the case of multi-selection list boxes. Otherwise the existing selection is left intact.

<b>Applies to</b>	List box control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## BACKGROUND-COLOUR-NAME

Provides a choice of existing background colors.

When setting background colors, you have four possibilities:

- Use the BACKGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). The color will then be determined by your color settings in the operating system.
- Use the BACKGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Define your own color by using the BACKGROUND-COLOUR-VALUE attribute in a dialog editor attributes window. To do so, you must first set the BACKGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Set the color dynamically in event handler code by assigning a value to the attribute.

### Note:

Under Windows NT Version 4.0, the push button control's background color will be displayed as the system default at runtime, regardless of the BACKGROUND-COLOUR-NAME value.

<b>Applies to</b>	Canvas control, control box control dialog (window, MDI frame, MDI child), edit area control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, table control, text constant control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	DEFAULT (0)
<b>Possible Values</b>	See table below.

### Possible Values

DEFAULT (0)	WHITE (1)	BLACK (2)
LTGRAY (3)	GRAY (4)	DKGRAY (5)
RED (6)	GREEN (7)	BLUE (8)
CYAN (9)	MAGENTA (10)	BROWN (11)
YELLOW (12)	LIGHTRED (13)	LIGHTGREEN (14)
LIGHTBLUE (15)	LIGHTCYAN (16)	LIGHTMAGENTA (17)
BRIGHTWHITE (18)	CUSTOM (50)	

### Note:

The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

## BACKGROUND-COLOUR-VALUE

Provides a facility to define customized background colors.

When setting background colors, you have four possibilities:

- Define your own color by using the BACKGROUND-COLOUR-VALUE attribute in a dialog editor Attributes window. To do so, you must first set the BACKGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Use the BACKGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). Your color will then be determined by your color settings in the operating system.
- Use the BACKGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Set the color dynamically in event handler code by assigning a set of three RGB values to the attribute. To do so, you must first set the BACKGROUND-COLOUR-NAME attribute to CUSTOM (50). You enter the three RGB values in hexadecimal form, such as "H'FF0000'.

### Note:

Under Windows NT Version 4.0, the push button control's background color will be displayed as the system default at runtime, regardless of the BACKGROUND-COLOUR-VALUE value.

<b>Applies to</b>	Canvas control, control box control dialog (window, MDI frame, MDI child), edit area control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, table control, text constant control, toggle button control.
<b>Data Type</b>	B3
<b>Default Value</b>	None
<b>Possible Values</b>	Any red, green, and blue value from 1 to 253

## BAR-ID

Specifies an identifier for a status bar or tool bar control. The value is used to identify the control between sessions when the bar layout is saved and restored via the SAVE-LAYOUT and LOAD-LAYOUT actions, and must therefore be unique within a dialog.

<b>Applies to</b>	Status bar control, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 to 250

## BITMAP-FILE-NAME

Assigns a bitmap file to a dialog element or assigns an icon to a dialog. If no path information is supplied, the bitmap is searched for in the logon library's RES subdirectory first, then in the RES subdirectory of each steplib, then in the directory assigned to the environment variable NATGUI\_BMP. If no bitmap file with this name exists in any of the search paths, the file "default.bmp" in the NATGUI\_BMP directory is assigned.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child), bitmap control, menu item, signal, status bar pane, tool bar item.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string, or file name of up to 253 alphanumeric characters

## CELL-ATTRIBUTES

Determines whether the attributes FOREGROUND-COLOUR-NAME, BACKGROUND-COLOUR-NAME, FOREGROUND-COLOUR-VALUE, BACKGROUND-COLOUR-VALUE and DIL-TEXT of the table control apply to the whole table (FALSE) or to a single cell of the table (TRUE). If the above attributes apply to a cell, the ROW and COLUMN attribute of the table specify which cell is changed.

<b>Applies to</b>	Table Control
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## CHECKED

Determines whether a dialog element is checked (selected) or not.

<b>Applies to</b>	Menu item, radio button control, signal, table control, toggle button control, tool bar item.
<b>Data Type</b>	I4
<b>Default Value</b>	UNCHECKED
<b>Possible Values</b>	UNCHECKED / CHECKED

## CLIENT-DATA

Data associated with a dialog element. Often, it is useful to store information such as a database key or an array index with a dialog element. This allows for quick retrieval of the client data. If you use this attribute with a dialog (window, MDI frame, MDI child), Natural automatically assigns the dialog ID and you may therefore only query this attribute.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, context menu, column specification control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, input field control, line control, list box control, list box item, menu bar, menu item, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, selection-box item, signal, status bar control, status bar pane, submenu control, table control, timer, tool bar control, toggle button control, tool bar, tool bar item.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	Any I4 value

## CLIENT-HANDLE

Specifies a general-purpose attribute that can be used to store the handle of any GUI object. This attribute is not used by Natural itself.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any dialog or dialog element handle

## CLIENT-KEY

Alphanumeric data associated with a dialog or a dialog element. Often, it is useful to store a text string with a dialog element to allow for direct access of the string. The CLIENT-DATA attribute enables you to assign an integer value to many dialog elements; the CLIENT-KEY attribute is used in conjunction with the attribute CLIENT-VALUE and enables you to store alphanumeric key/string pairs.

You first assign a value to the CLIENT-KEY attribute. This determines the key under which the string is to be stored for an instance of a dialog element. You then assign an alphanumeric string to the CLIENT-VALUE attribute of the dialog element.

It is advisable to reuse keys that are not needed because you may use only a limited number of keys.

You can query the CLIENT-VALUE of a dialog element by first setting the CLIENT-KEY attribute of the dialog element to the desired value and then accessing the CLIENT-VALUE attribute.

**Note:**

You cannot query the CLIENT-KEY for any type of dialog element (GUI control) because there can be several keys for one dialog element and it would not be clear which one should be queried in the particular context. To avoid this, you must know which CLIENT-KEY value your dialog element is using.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string (no key/value pair will be assigned to the dialog element)
<b>Possible Values</b>	Empty string or any text string

## CLIENT-VALUE

Alphanumeric data associated with a dialog or a dialog element. For a description of the context in which this attribute is used, see the attribute CLIENT-KEY.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string (no key/value pair will be assigned to the dialog element)
<b>Possible Values</b>	Empty string or any text string

## COLUMN

A COLUMN attribute value corresponds to exactly one ROW attribute value. The combination of these two denotes a cell inside a table control.

- To denote a cell of the table: COLUMN must be  $\geq 1$  and ROW must be  $\geq 1$ .
- To denote the first column in the table: COLUMN must be 0 and ROW must be 1.
- To denote the header row in the table: COLUMN must be 1 and ROW must be 0.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## COLUMN-COUNT

Enables you to query the current number of columns in a table control. This table is set internally on creation of the table control. It cannot be changed.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## COLUMN-TYPE

Determines the type of column in a column specification control (which itself is part of a table control). If, for example, the value is set to INPUTFIELD, all cells in this column behave as if they were input field controls.

<b>Applies to</b>	Column specification control.
<b>Data Type</b>	I4
<b>Default Value</b>	INPUTFIELD
<b>Possible Values</b>	INPUTFIELD / SELECTIONBOX / TOGGLEBUTTON

## CONTEXT-MENU

Specifies the handle of the context menu associated with the control. A value of NULL-HANDLE indicates that the control has no context menu assigned. The context menu specified here overrides any default context menu displayed by the system. Note that, for table controls, the specified context menu is not displayed for a cell which is being edited. For MDI frames, the context menu applies to the MDI client area.

<b>Applies to</b>	ActiveX control, bitmap control, canvas control, dialog (window, MDI frame, MDI child), edit area control, input field control, list box control, push button control, radio button control, scroll bar control, selection box control, status bar control, table control, toggle button control, tool bar control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / handle value of context menu.

## DEFAULT-BUTTON

If you assign the handle value of a push button control to the dialog's DEFAULT-BUTTON attribute, the push button control's click event handler is triggered when the end user presses ENTER in the dialog. This is true unless another push button control has the focus, in which case pressing of ENTER triggers the focused button's click event handler. A push button control defined as DEFAULT-BUTTON overrides a push button control defined as OK button (STYLE = 'O'). Alternatively to a push button also an ActiveX control can be assigned to the DEFAULT\_BUTTON attribute, provided it behaves like a button. These ActiveX controls are marked in the system registry with the style OLEMISC\_ACTSLIKEBUTTON.

**Note:**

Default buttons have a black margin.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE or any push button control handle

## DIL-TEXT

Determines the DIL text for a dialog element. The dynamic information line and the status line are updated with this text when this dialog element gets the focus. However, the text will only be displayed if HAS-DIL is TRUE. Then the DIL-TEXT will overlap the STATUS-TEXT value or vice versa, depending on which was modified last.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, column specification control, edit area control, input field control, list box control, menu item, OLE container control, push button control, radio button control, scroll bar control, selection box control, signal, table control, toggle button control, tool bar item.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string / any text

## DOCKING

Specifies the sides of the owning dialog on which docking is allowed. In order for a dockable tool bar control to be able to dock, both the tool bar control and the owning dialog must allow docking on the specified side(s). The supported docking positions are: none (tool bar control may only be floated), top only, bottom only, left only, right only, horizontal (top or bottom), vertical (left or right) or any side.

<b>Applies to</b>	Dialog (window, MDI Frame), tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	DL-ANY
<b>Possible Values</b>	DL-NONE / DL-LEFT / DL-TOP / DL-RIGHT / DL-BOTTOM / DL-VERT / DL-HORZ / DL-ANY

## DPI

Specifies the number of dots (pixels) per logical inch in effect at the time the dialog was last saved from within the editor. This attribute, if non-zero, also indicates that the dialog is scalable.

**Note:**

The value of this attribute varies depending on whether small or large fonts (or a custom font size setting) are being used.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 200

## DRAGGABLE

Determines whether the end user can use the mouse to drag a bitmap control onto another bitmap control in the same dialog. When dropping the bitmap control by releasing the mouse button, a drag and drop event will occur for the target bitmap control. Whenever a bitmap control is DRAGGABLE, you can also use the INQ-DRAG-DROP action provided with the PROCESS GUI statement to find out where the bitmap control was dropped.

<b>Applies to</b>	Bitmap control, tool bar control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## EDIT-MASK

Specifies a Natural edit mask for a dialog element that requires end user input. (An edit mask determines character by character the format in which the field values are to be entered and displayed.) You use this attribute to validate end user input. You can, however, only validate input if you use the "Linked Variable" option in the Attribute Source dialog box, which can be accessed by clicking on the STRING's "Source" button in the dialog element's Attributes window. When the dialog element loses the focus (the end user leaves the input-field), the content of the field is checked against the edit mask. If the input is not compatible with the edit mask, Natural displays a message box asking the end user to "Retry" or "Cancel".

<b>Applies to</b>	Input field control, selection box control.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string / any Natural edit mask

## EMBEDDED-OBJECT

The value of this attribute is the name of a Natural embedded object, which has the default file extension ".neo". When a value is assigned to an OLE container control's EMBEDDED-OBJECT attribute, Natural loads the embedded object into the container, discarding any previously loaded object. If the values of the SERVER-OBJECT and SERVER-PROGID attributes are set for the OLE container control, the value of the EMBEDDED-OBJECT attribute is removed.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	A253
<b>Default Value</b>	None
<b>Possible Values</b>	All file names (if a file name is specified without an extension, the extension ".neo" is added by default)

## ENABLED

Determines whether the end user may use the dialog element (for input, for example) or whether the dialog element is disabled (greyed, for example).

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, context menu, control box control dialog (window, MDI frame, MDI child), edit area control, group frame control, input field control, list box control, menu item, OLE container control, push button control, radio button control, scroll bar control, selection box control, signal, status bar control, status bar pane, table control, toggle button control, tool bar control, tool bar item.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## EVENT-QUEUEING

Determines whether messages received from the windowing system for this dialog should be processed immediately. By default, certain messages are queued in order to be compatible with the behaviour of older Natural versions under Windows 3.x. This attribute allows event queueing to be disabled at the dialog level, causing these messages to be processed immediately.

Event queueing was introduced under Windows 3.x to serialize messages dispatched via nested message loops. Such nested message loops were common under Windows 3.x in order to wait for a potentially lengthy operation (such as a remote database access) without hanging Windows. This is no longer a problem under newer versions of Windows, because these are pre-emptive operating systems. As such, it is no longer possible for an application that is not processing its messages to hang the system. Therefore, it is usually preferable (especially for newly written applications) to turn event queueing off.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## FIRST-CHILD

The FIRST-CHILD attribute serves as a tool to query the first created child dialog element, that is, the dialog element one level below the (parent) dialog element. An example of such a child-parent relationship between dialog elements is the relationship between a dialog and several push button controls inside. With the FIRST-CHILD attribute, you query the push button control that has been created first inside the dialog. You use this attribute in conjunction with the SUCCESSOR attribute to "travel" through the hierarchy of dialog elements.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE (no child existing) / any dialog element handle

## FIRST-COLUMN-WIDTH

Determines the width in pixels of the first column in a table control. The attribute is used if the HAS-FIRST-COLUMN attribute is TRUE for the table control.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	50
<b>Possible Values</b>	0 - 9999

## FIRST-VISIBLE-COLUMN

Determines the index of the first visible column.

- If no frozen columns are defined, the first visible column is the leftmost column of the table.
- If frozen columns *are* defined, the first visible column is the first column displayed to the right of the frozen columns.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## FIRST-VISIBLE-ROW

Determines the index of the row which is displayed at the top of the table.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## FOLLOWS

Specifies the preceding dialog element in the (new) control sequence. The control sequence determines the order in which dialog elements receive the focus on repeated pressing of the TAB key (or arrow keys, for controls with the same group ID). It also determines which dialog element gets the focus on input of a mnemonic corresponding to a dialog element which cannot receive the focus itself (for example, a text-constant or group frame). By default, the FOLLOWS attribute is based on the SUCCESSOR chain. You can use this attribute to move a dialog element from its default position. Assigning a value of NULL-HANDLE moves the dialog element to the end of the control sequence, whereas assigning the handle of the parent dialog itself moves the dialog to the front of the control sequence. Assigning the handle of the control itself has the effect of removing the control from the control sequence such that it can no longer be reached via the TAB key.

When applied to dialogs, the FOLLOWS attribute returns the last dialog element in the control sequence. This enables complete enumeration of the control sequence in reverse order.

<b>Applies to</b>	ActiveX control, bitmap control, canvas control, control box control, dialog (window, MDI frame, MDI child), edit area control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, scroll bar control, selection box control, status bar control, table control, text constant control, toggle button control, tool bar control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / handle value of the desired predecessor in the control sequence.

## FONT-HANDLE

Serves as a handle to assign a certain combination of font face, font size, and font style to a dialog element. Assigning an existing font handle value to this attribute leads to the STRING of the dialog element being displayed in this font. You can use the same FONT-HANDLE for several dialog elements.

<b>Applies to</b>	Edit area control, graphic text control, group frame control, input field control, list box control, push button control, radio button control, selection box control, status bar control, table control, text constant control, toggle button control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	System font
<b>Possible Values</b>	System font / any other defined fonts

# FONT-STRING

Sets the default font throughout a dialog. The default font must be set as a parameter while the dialog is being created with PROCESS GUI action ADD. If a dialog element inside the dialog contains text with no particular font assigned to it, this text will be displayed in the font specified by FONT-STRING. When you assign a FONT-STRING to a dialog, a font control is created internally, with the dialog being the parent of the font control. The font's handle can be queried from the dialog's FONT-HANDLE attribute.

**Note:**

If the FONT-STRING attribute has created fonts that are too large or too small for the dialog window layout, you can use this attribute to resize all fonts evenly throughout the dialog.

<b>Applies to</b>	Dialog.
<b>Data Type</b>	A253
<b>Default Value</b>	System font
<b>Possible Values</b>	Encoded string which can be edited in the dialog editor's dialog attributes window. Font strings consist of several fields separated by a delimiter character.

**Possible Delimiter Characters**

The default delimiter character is the slash "/".

Possible delimiter characters are:

"/", ",", "#", ":", ";", "|", "@", "\$", "%", "&".

**Fields**

1. The font's face name.
2. The font style: Empty, or equivalently, the word "Regular", or a combination of the keywords "Bold", "Italic", "Strikeout", and "Underlined", separated by blanks.
3. The font size: Either the size in points (here, an optional decimal point followed by one digit is allowed), or the size as "*width x height*". A positive value indicates character size, a negative value indicates cell size. A zero indicates that the system should choose a value for the field. The "*width x height*" notation allows for better portability across screen resolutions.
4. Character set information. A combination of keywords, one from each of the following subgroups, separated by blanks:

<b>Character sets:</b>	ANSI or SYMBOL or SHIFTJIS or OEM.
<b>Fixed or variable-width font:</b>	FIXED or VARIABLE.
<b>Font family:</b>	ROMAN or SCRIPT or SWISS or MODERN or DECORATIVE.
<b>Print quality:</b>	DRAFT or PROOF.

5. A character indicating the platform. For fonts selected under windows, this character is "W".
6. Platform-dependent field. For Windows, this is the clipping precision.
7. Platform-dependent field. For Windows, this is the output precision.

**Attributes**

**FONT-STRING**

**Sample String**

**/Arial/Bold/0 x -19/ ANSI VARIABLE SWISS DRAFT/W/2/0/**

## FOREGROUND-COLOUR-NAME

Provides a choice of existing foreground colors. Foreground colors also refer to text.

When setting foreground colors, you have four possibilities:

- Use the FOREGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). The color will then be determined by your color settings in the operating system.
- Use the FOREGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Define your own color by using the FOREGROUND-COLOUR-VALUE attribute in a dialog editor attributes window. Here, you must first set the FOREGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Set the color dynamically in the event handler code by assigning a value to the attribute.

**Note:**

Under Windows NT Version 4.0 and under Windows 95, a push button control's foreground color will be displayed as the system default at runtime, regardless of the FOREGROUND-COLOUR-NAME value.

<b>Applies to</b>	Canvas control, edit area control, graphic text control, group frame control, input field control, line control, push button control, radio button control, rectangle control, selection box control, table control, text constant control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	DEFAULT (0)

**Possible Values**

DEFAULT (0)	WHITE (1)	BLACK (2)
LTGRAY (3)	GRAY (4)	DKGRAY (5)
RED (6)	GREEN (7)	BLUE (8)
CYAN (9)	MAGENTA (10)	BROWN (11)
YELLOW (12)	LIGHTRED (13)	LIGHTGREEN (14)
LIGHTBLUE (15)	LIGHTCYAN (16)	LIGHTMAGENTA (17)
BRIGHTWHITE (18)	CUSTOM (50)	

**Note:**

The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

## FOREGROUND-COLOUR-VALUE

Provides a facility to define customized foreground colors. Foreground colors also refer to text.

When setting foreground colors, you have four possibilities:

- Define your own color by using the FOREGROUND-COLOUR-VALUE attribute in a dialog editor Attributes window. Here, you must first set the FOREGROUND-COLOUR-NAME attribute to CUSTOM (50). A dialog box appears where you can set values for the red, green, and blue elements of your color (RGB model).
- Use the FOREGROUND-COLOUR-NAME attribute and leave the value at DEFAULT (0). Your color will then be determined by your color settings in the operating system.
- Use the FOREGROUND-COLOUR-NAME attribute and choose from one of the dialog editor's predefined colors. These are represented by the values 1 to 18.
- Set the color dynamically in event handler code by assigning a set of three RGB values to the attribute. To do so, you must first set the FOREGROUND-COLOUR-NAME attribute to CUSTOM (50). You enter the three RGB values in hexadecimal form, such as "H'FF0000'.

### Note:

Under Windows NT Version 4.0 and under Windows 95, a push button control's foreground color will be displayed as the system default at runtime, regardless of the FOREGROUND-COLOUR-VALUE value.

<b>Applies to</b>	Canvas control, edit area control, graphic text control, group frame control, input field control, line control, push button control, radio button control, rectangle control, selection box control, table control, text constant control, toggle button control.
<b>Data Type</b>	B3
<b>Default Value</b>	None
<b>Possible Values</b>	Any red, green, and blue value from 1 to 253

## FROZEN-COLUMNS

Determines the number of columns on the left side of the table control that cannot be scrolled horizontally.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## GROUP-ID

Assigns a group ID to a radio button control. Radio button controls with the same GROUP-ID value are regarded as a group; that is, only one radio button control out of this group may be CHECKED at any time.

<b>Applies to</b>	Radio button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any other integer value

## HANDLE-VARIABLE

This attribute is only used in the context of the PROCESS GUI statement action ADD. It identifies the Natural variable which receives the handle of the newly created dialog element. Note that the value of this attribute is set at creation time of the corresponding dialog element; it cannot be queried or modified.

<b>Data Type</b>	HANDLE
<b>Possible Values</b>	Any Natural variable

### Example:

```
DEFINE DATA LOCAL 1
#NEW2 HANDLE OF INPUTFIELD END-DEFINE
...
PROCESS GUI ACTION ADD WITH PARAMETERS HANDLE-VARIABLE
= #NEW2 TYPE = INPUTFIELD STRING
= 'NEW2' RECTANGLE-X
= 24 RECTANGLE-Y
= 30 RECTANGLE-W
= 176 RECTANGLE-H
= 28 ENABLED
= TRUE VISIBLE
= TRUE PARENT
= #DLG$WINDOW
END-PARAMETERS
```

## HAS-DIL

Determines whether a status bar control or the status line of a dialog is updated with DIL-TEXTs. When applied to a dialog, the following interdependencies exist for the display of DIL-TEXTs and STATUS-TEXTs:

HAS-DIL	HAS-STATUS-BAR	DIL-TEXT	STATUS-TEXT
TRUE	TRUE	displayed	displayed
TRUE	FALSE	-	-
FALSE	TRUE	-	displayed
FALSE	FALSE	-	-

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), status bar control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HAS-FIRST-COLUMN

Determines whether the table control displays a first column to the left of the other columns. This first column consists of a button series to the left of each row. Each button is inscribed with the number of the row.

<b>Applies to</b>	Table control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## HAS-HELP-BUTTON

Determines whether the dialog title bar has a help button or not. Selection of this button invokes help mode. You can then display help for a control simply by clicking on it with the left mouse button.

**Note:**

Even if this attribute is set to TRUE, Windows will not display the help button if the maximize and minimize buttons are also present.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HAS-MENU-BAR

Indicates whether the dialog has a menu bar.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HAS-STATUS-BAR

Determines whether a status bar is displayed at the bottom of the dialog window. This status bar displays the value of the STATUS-TEXT attribute or, if applicable, of the DIL-TEXT attribute of the dialog element that currently has the focus. For the display of DIL-TEXTs and STATUS-TEXTs, the following interdependencies exist:

HAS-DIL	HAS-STATUS-BAR	DIL-TEXT	STATUS-TEXT
TRUE	TRUE	displayed	displayed
TRUE	FALSE	-	-
FALSE	TRUE	-	displayed
FALSE	FALSE	-	-

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HAS-SYSTEM-BUTTON

Determines whether the dialog has a system button or not. The system button is in the top left corner of the window and provides a menu with the platform's standard window functions (like Close or Minimize).

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## HAS-TOOLBAR

Determines whether the tool bar of a dialog (that is, the tool bar plus the tool bar items) is to be displayed or not. You use this attribute to dynamically switch the tool bar on and off. Note that the handle value of the tool bar must be assigned to the TOOLBAR-HANDLE attribute.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HAS-TOOLTIP

Indicates whether tool tips are to be displayed for a tool bar or status bar control. If this attribute is set to FALSE, tool tip display is suppressed.

<b>Applies to</b>	Status-bar control, tool bar control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## HEADER-FONT-HANDLE

Serves as a handle to assign a certain combination of font face, font size, and font style to the first column and the header line of a table control. Assigning an existing font handle value to this attribute leads to the header being displayed in this font.

<b>Applies to</b>	Table control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	System font
<b>Possible Values</b>	System font / any other defined fonts

## HEADER-HEIGHT

Determines the height (in pixels) of the header line in a table control. To add a header line, the table control's STYLE attribute must include the letter "h".

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## HELP-FILENAME

Determines a dialog's help file name usually without extension. It is assumed that the extension is ".chm" or ".hlp". If the end user presses F1 in an active dialog, Natural first queries for a file with the value of HELP-FILENAME plus the extension ".chm". If it does not find such a file, it queries for a file with the value of HELP-FILENAME plus the extension ".hlp". If HELP-FILENAME is not specified for the current dialog, the HELP-FILENAME of the parent dialog (if any) is used, if specified. This process is repeated as necessary until a top-level dialog is reached. If no non-empty HELP-FILENAME attribute is found for any of these dialogs, Natural searches for the file *libraryname.chm* first, then (if not found) *libraryname.hlp*, where *libraryname* is the name of the logon library.

The HELP-FILENAME attribute can also contain path information. If this is the case, then only that path is searched. If no path information is supplied, the help file is searched for in the logon library's RES subdirectory first, then in the RES subdirectory of each steplib, then in the directory assigned to the environment variable NATGUI\_BMP.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	A253
<b>Default Value</b>	None
<b>Possible Values</b>	Any file name

## HELP-ID

The help ID to be passed to the help system whenever the end user requests help for the dialog or dialog element that has the focus by pressing F1. In most help systems, this help ID is referred to as topic number.

If the dialog or dialog element with the focus does not provide a HELP-ID, the HELP-ID of its parent is taken.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, column specification control, context menu dialog (window, MDI frame, MDI child), edit area control, input field control, list box control, menu bar, OLE container control, push button control, radio button control, scroll bar control, selection box control, submenu control, table control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any other integer value

## HORIZ-SCROLLABLE

Determines whether the dialog or dialog element has a horizontal scroll bar or not.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), edit area control, table control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## ICONIZED

Determines whether the OLE container control represents data visually inside the control's borders or whether data are displayed as the server application's icon (useful for sound or video).

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## INPLACE-ACTIVE

Determines whether the server of the OLE container control is in-place active or not.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## ITEM-H

Determines the height of all items in the tool bar. For status bar controls, this is the minimum pane height.

<b>Applies to</b>	Status bar control, tool bar, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (system default)
<b>Possible Values</b>	0 - 9999

## ITEM-W

Determines the width of all items in the tool bar.

<b>Applies to</b>	Tool bar, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (system default)
<b>Possible Values</b>	0 - 9999

## LAST-CHILD

Contains the handle of the dialog element's last child dialog element, or NULL-HANDLE if the dialog element has no children. You can use this attribute in conjunction with the PREDECESSOR and PARENT attributes to traverse the parent/child hierarchy in the reverse direction.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE (no child existing) / any dialog element handle.

## LENGTH

Specifies the maximal length of a string to be entered by the end user. If you have set an EDIT-MASK attribute for the same dialog element, the edit mask overrides the LENGTH specified here.

<b>Applies to</b>	Column specification control, edit area control, input field control, selection box control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (= unlimited) 253 for input field controls and selection box controls. Maximum system limits for edit area controls.
<b>Possible Values</b>	0 - 30000 for edit area controls, or 0 - 253 for input field controls and selection box controls (and column specification controls based on these two types).

## LINE

Determines the number of logical units by which a scroll bar control's slider moves if the end user presses the UP ARROW and DOWN ARROW buttons.

Inside a numeric scale limited by the values of the scroll bar control's MIN and MAX attributes, the value of this attribute determines the size of the steps by which the slider will be moved. If you set a small value in relation to the MIN and MAX attributes, the slider will move continuously. If you set a larger value, the slider will move in visible steps.

<b>Applies to</b>	Scrollbar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any integer value

## LINKED

Determines whether the variable specified for the dialog element is linked or not. Setting this attribute to TRUE and then assigning a Natural variable to the VARIABLE attribute has the same effect as specifying a "Linked Variable" in a source dialog box of a "String" attribute window.

<b>Applies to</b>	Input field control, selection box control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

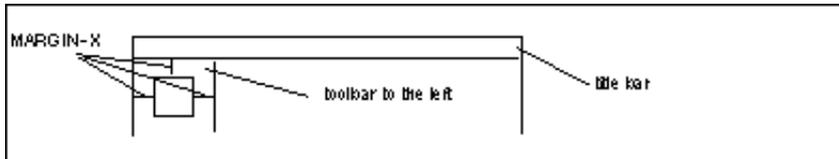
## LOCATION

Specifies the position (top, bottom, left, right) of a status bar or tool bar control relative to the associated dialog. For dockable tool bars floated in a separate window, the location is represented by the value TB-FLOAT. Status-bar controls may only be positioned at the top or bottom of a dialog.

<b>Applies to</b>	Status-bar control, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	TB-TOP
<b>Possible Values</b>	TB-TOP / TB-BOTTOM / TB-LEFT / TB-RIGHT / TB-FLOAT

## MARGIN-X

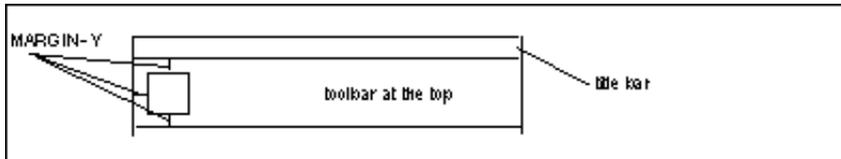
Determines which margin to the left, to the right and above the tool bar items is displayed in the tool bar area. This attribute only applies if TOOLBAR-POS is set to TB-LEFT or TB-RIGHT.



<b>Applies to</b>	Status bar control, tool bar, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## MARGIN-Y

Determines which margin to the left, above and below the tool bar items is displayed in the tool bar area. This attribute only applies if TOOLBAR-POS is set to TB-TOP or TB-BOTTOM.



<b>Applies to</b>	Status bar control, tool bar, tool bar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## MAX

Determines the maximum value on a scroll bar control's (numeric) scale. When using a scroll bar control, you must always determine values for the MIN and MAX attributes to set limits to the scale.

**Note:**

The difference between the values of the MAX and MIN attributes must be less than 32767.

<b>Applies to</b>	Scrollbar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any integer value between -32767 and 32767; must be larger than the value of the MIN attribute.

## MAXIMIZABLE

Determines whether the dialog has a maximize button in the top right corner. This button enables you to maximize the dialog so that it covers the whole screen.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## MAXIMIZED

If this attribute is set, the dialog is maximized to full screen size and the dialog's size event occurs. If you have specified a size event handler code section, this code section will be triggered. An MDI child window will be maximized to fill the MDI frame window's client area.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## MENU-HANDLE

When used with a dialog, it creates a link between a menu bar plus its submenu controls, its items, and this dialog. The menu is then displayed at the top of the dialog. When the MENU-HANDLE attribute is modified to NULL-HANDLE, the menu disappears without being destroyed. When used with a menu item, it creates a link between a submenu control and this menu item. The menu structure is then nested by one more level. Several dialogs may be linked to the same menu bar, but you may only link one menu bar to a dialog.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), menu item.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any handle of a menu bar

## MENU-ITEM-OLE

Determines whether a top-level menu item or a submenu control represents the OLE Container or File or Window group or not. If a dialog has a menu bar and an OLE container control is being edited in-place, the submenus of the OLE server application are merged into the Natural menu bar. The Natural contribution to the merged menu consists of the submenu controls called File, Container, and Window, depending on what you set with this attribute. The server application's contribution are the submenu controls called Edit, Object, and Help.

<b>Applies to</b>	Menu item, submenu control.
<b>Data Type</b>	I4
<b>Default Value</b>	MO-NORMAL

### Possible Values

<b>MO-NORMAL</b>	No OLE menu item.
<b>MO-CONTAINER</b>	OLE Container group.
<b>MO-FILE</b>	OLE File group.
<b>MO-WINDOW</b>	OLE Window group.

## MENU-ITEM-TYPE

Determines the function of a menu item.

The values MT-CUT, MT-COPY, MT-PASTE, MT-DELETE and MT-UNDO are predefined functions that are executed by the windowing system. They are automatically enabled or disabled depending on the focus and on the selection.

<b>Applies to</b>	Menu item, signal, tool bar item.
<b>Data Type</b>	I4
<b>Default Value</b>	MT-NORMAL

### Possible Values

<b>MT-NORMAL</b>	This is a menu item on the lowest level of the menu structure. Only items of this type can have an event handler.
<b>MT-SEPARATOR</b>	Items of this type are displayed as horizontal line in a submenu to separate menu items optically.
<b>MT-CUT</b>	Items of this type allow the end user to cut a selected portion of text and copy it to the clipboard. (The cut text is in the dialog element that currently has the focus).
<b>MT-COPY</b>	Items of this type allow the end user to copy a selected portion of text to the clipboard. (The copied text is in the dialog element that currently has the focus).
<b>MT-PASTE</b>	Items of this type allow the end user to paste the portion of text that is in the clipboard. (The text is pasted into the dialog element that currently has the focus).
<b>MT-DELETE</b>	Items of this type allow the end user to delete a selected portion of text. (The deleted text is in the dialog element that currently has the focus).
<b>MT-UNDO</b>	Items of this type allow the end user to undo a text input operation. The previous text input operation is undone in the dialog element that currently has the focus.
<b>MT-SUBMENU</b>	Items of this type represent a submenu. When selected, a (*) submenu drops down.
<b>MT-WINDOWMENU</b>	Items of this type also represent a submenu, only that all MDI (*) children are entered automatically into it.
<b>MT-MDITILE</b>	Arranges the MDI children of an MDI frame dialog in a tile (**) manner.
<b>MT-MDICASCADE</b>	Arranges the MDI children of an MDI frame dialog in a cascade (**) manner.
<b>MT-MDIARRANGE (**)</b>	Arranges the iconized MDI children of an MDI frame dialog.
<b>MT-OBJECTVERBS</b>	Displays and enables the verbs (actions) available for an OLE container control in the current dialog if the OLE container control has the focus and is assigned to an OLE server application. If the focus is on another dialog element, the disabled entry "Object" appears.

\* If you create a menu item of this type, you must assign the handle value of the submenu control to the MENU-HANDLE attribute.

\*\* Menu items of this type must be contained in a menu item of type MT-WINDOWMENU.

### Note:

The local data area NGULKEY1 in library SYSTEM lists reserved keywords to be used in any event handler code. This enables you to refer to the above attribute values by the more meaningful keywords rather than by the

IDs. It also enables you to use meaningful dialog element names as parameters in a CALLNAT to an NGU-prefixed subprogram or in an OPEN DIALOG to an NGU-prefixed Dialog or in a PROCESS GUI statement action.

## MIN

Determines the minimum value on a scroll bar control's (numeric) scale. When using a scroll bar control, you must always determine values for the MIN and MAX attributes to set limits to the scale.

**Note:**

The difference between the values of the MAX and MIN attributes must be less than 32767.

<b>Applies to</b>	Scrollbar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any integer value between -32767 and 32767; must be smaller than the value of the MAX attribute.

## MINIMIZABLE

Determines whether the dialog has a minimize button in the top right corner. This button enables you to minimize the dialog to an icon.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## MINIMIZED

If this attribute is set, the dialog is minimized and appears as an icon and the dialog's size event occurs. If you have specified a size event handler code section, this code section will be triggered.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## MODIFIABLE

Determines whether the content in a dialog element can be edited or not. If the attribute's value is set to TRUE, the dialog element's text can be edited; if the value is set to FALSE, the dialog element can only be used for output (but it remains scrollable, as opposed to ENABLED = FALSE). If MODIFIABLE is FALSE for an OLE container control that has been activated for in-place editing, the OLE object can not be modified.

<b>Applies to</b>	Column specification control, edit area control, input field control, OLE container control, selection box control, table control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## MODIFIED

If set to TRUE, the content of a dialog element (e.g., *STRING* attribute for an input field control, *CHECKED* attribute for a toggle button control, etc.) has been modified since the last resetting of the *MODIFIED* attribute. It may be modified either directly by user interaction or indirectly by an event handler: either the open event handler resets it indirectly on opening the dialog or the user resets it directly in the attributes window of the dialog that contains the dialog element in question.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), edit area control, input field control, list box control, radio button control, scroll bar control, selection box control, table control, toggle button control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## MODIFIED-SUCCESSOR

Gets all modified dialog elements of a dialog one after the other. To get the first modified dialog element in a dialog, you use the handle of the dialog to query the attribute value. To get the next modified dialog element, you use the handle of the dialog element queried last. If no more modified dialog elements are found, a NULL-HANDLE is returned and the query for a modified successor is ended. If you query the MODIFIED-SUCCESSOR of a dialog (window, MDI frame, MDI child), the handle value of the dialog's first modified child will be returned.

To get the selected items of a list box control, you use the SELECTED-SUCCESSOR attribute.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), edit area control, input field control, list box control, radio button control, scroll bar control, selection box control, table control, toggle button control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any dialog element handle

## MULTI-SELECTION

Determines whether a list box control offers a multiple selection or not. If set to TRUE, you can get the selected items using the SELECTED-SUCCESSOR attribute.

<b>Applies to</b>	List box control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## NAME

Specifies the dialog's object name, without the ".NS3" file extension.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	A8
<b>Default Value</b>	None
<b>Possible Values</b>	Name of dialog object.

## OBJECT-SIZE

Determines the size of the object in the OLE container control.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any other integer value.

## OFFSET-X

Determines the x-axis offset on a canvas control. If, for example, a line control that was created on a canvas control exceeds the size of the canvas control, the line is clipped and the exceeding rectangular area is not displayed. The OFFSET-X attribute determines the x-axis offset of the canvas control's upper border against the upper border of the exceeding area.

<b>Applies to</b>	Canvas control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## OFFSET-Y

Determines the y-axis offset on a canvas control. If, for example, a line control that was created on a canvas control exceeds the size of the canvas control, the line is clipped and the exceeding rectangular area is not displayed. The OFFSET-Y attribute determines the y-axis offset of the canvas control's upper border against the upper border of the exceeding area.

<b>Applies to</b>	Canvas control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## P1-X

Determines a line control's start position on the x-axis.

<b>Applies to</b>	Line control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## P1-Y

Determines a line control's start position on the y-axis.

<b>Applies to</b>	Line control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## P2-X

Determines a line control's end (target) position on the x-axis.

<b>Applies to</b>	Line control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## P2-Y

Determines a line control's end (target) position on the y-axis.

<b>Applies to</b>	Line control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## PAGE

Determines the number of logical units by which a scroll bar control's slider moves if the end user clicks inside the scroll bar control's shaft.

Within a numeric scale limited by the values of the scroll bar control's MIN and MAX attributes, the value of this attribute determines the size of the steps by which the slider will be moved. If you set a small value in relation to the MIN and MAX attributes, the slider will move continuously. If you set a larger value, the slider will move in visible steps.

<b>Applies to</b>	Scrollbar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any other integer value

## PARENT

Determines the parent handle of a dialog or a dialog element in the hierarchy. At the top of the hierarchy are one or several base dialogs; these are the parents of the dialog elements contained in them. These children may themselves contain children, such as controls in a child dialog. Children on the same level are called siblings. They may be enumerated using the SUCCESSOR attribute with the same parent.

The parent/child relationship can only be changed by "deleting" a child and creating a new one.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any dialog element

## POPUP-HELP

Determines whether help invoked for the dialog or one of its controls via the title bar help button or pressing F1 should be displayed in a popup window. A popup window is a tool tip style window usually containing only text, which is automatically removed when the window loses the focus.

Help for popup windows can be created via the Help Organizer in the same way as for non-popup help. For HTML help, it is currently necessary to enter popup help text into a separate text file, which is then compiled into the help file together with the HTML topics.

**Note:**

To invoke the popup help in the latter case, it is necessary to explicitly specify the name of the text file in the HELP-FILENAME attribute via the syntax "*helpfilename.chm::/textfilename.txt*". Please refer to your HTML authoring tool help for further information.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## PREDECESSOR

Contains the handle of the previous dialog element with the same PARENT in the parent/child hierarchy, or NULL-HANDLE if no such dialog element exists. You can use this attribute to go through all dialog elements in one parent dialog or dialog element in reverse order.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / handle value of the predecessor.

## RECTANGLE-H

Represents the height of a dialog element.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, status bar control, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## RECTANGLE-W

Represents the width of a dialog element. If used with a column specification control, it represents the width of the column in pixels.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, column specification control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, status bar control, status bar pane, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## RECTANGLE-X

Represents the dialog element's location on the x-axis relative to the top left corner of its parent's client area.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, status bar control, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## RECTANGLE-Y

Represents the dialog element's location on the y-axis relative to the top left corner of its parent's client area.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, group frame control, input field control, list box control, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, status bar control, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## ROW

A ROW attribute value corresponds to exactly one COLUMN attribute value. The combination of these two denotes a cell inside a table control.

- To denote a cell of the table: COLUMN must be  $\geq 1$  and ROW must be  $\geq 1$ .
- To denote the first column in the table: COLUMN must be 0 and ROW must be 1.
- To denote the header row in the table: COLUMN must be 1 and ROW must be 0.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## ROW-COUNT

Enables you to query the current number of rows in a table control. Does not allow you to set or modify this number.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	1
<b>Possible Values</b>	0 - 9999

## ROW-HEIGHTx

Determines the height of all rows in a table control (except the header row).

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## SAME-AS

Specifies an equivalence relationship between two controls ("objects"), such that both objects are considered to represent the same user action ("command"). This allows Natural to ensure the coherency of the user interface implicitly. Coherency implies that all user interface elements representing the same command have the same state at all times. For example, if a menu item and a tool bar item represent the same command, then, at any one time, either both should be enabled, or both disabled. Furthermore, Natural ensures that the same event-handling code is executed when a command is invoked, regardless of which menu or tool bar item invoked it.

The referenced object may be a signal or a menu item. The referencing object may be a tool bar item or a menu item. A SAME-AS relationship between two menu items is not allowed. A two-step approach, whereby a tool bar item references a menu item, which itself references a signal, is permitted, but not recommended. Instead, the tool bar item should, like the menu item, reference the signal directly. Note that a SAME-AS relationship between objects belonging to different dialogs is permitted. Thus, objects in MDI child dialogs may reference objects in the MDI frame dialog.

For each command, the SAME-AS relationships form a dependency tree. The object within the tree that does not have a SAME-AS relationship to another object is the *root* object. Any object in the tree to which one or more SAME-AS relationships exist is referred to as a *referenced* object, and the objects which (directly or indirectly) reference it are referred to as that object's *dependent* objects.

- ACCELERATOR
- BACKGROUND-COLOUR-NAME
- BACKGROUND-COLOUR-VALUE
- BITMAP-FILE-NAME
- CHECKED
- DIL-TEXT
- ENABLED
- MENU-ITEM-TYPE
- SHARED
- TOOLTIP
- VISIBLE

Propagation implies that the same attribute is implicitly set to the same value for each object to which the attribute setting is propagated. Furthermore, when a SAME-AS relationship is set to any value other than NULL-HANDLE, the values of all the above attributes are immediately copied from the referenced object to the referencing object. Setting the SAME-AS attribute to NULL-HANDLE breaks the relationship and restores the object's creation-time attribute values.

In addition, a click received for a non-root object results in a CLICK event being raised for the root object instead. This ensures that the same code is executed regardless of how the command is accessed.

As an example, suppose a command may be accessed via a menu bar menu item, a context menu item, or a tool bar item. Instead of maintaining a set of state attributes for each of these three objects, it is easier to create a signal to represent the command, and to set the SAME-AS attribute of all three objects to the handle of this signal. Now, only the signal's attributes need to be maintained by the program. For example, setting the CHECKED attribute of the signal to TRUE implicitly also sets the CHECKED attribute of both menu items and the toolbar item to TRUE, thus checking them. Furthermore, regardless of which one of the menu or tool bar items the user clicks, a CLICK event is received for the signal instead.

<b>Applies to</b>	Menu item, tool bar item.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any signal or (for tool bar items only) menu item handle

## SCROLLRANGE-X

Determines the horizontal range within which the end user may scroll in a dialog window. The dialog editor automatically determines the scroll range so that it encloses all dialog elements. If the dialog's scroll bars are enabled, they are displayed if the dialog's size is smaller than the scroll range.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## SCROLLRANGE-Y

Determines the vertical range within which the end user may scroll in a dialog window. The dialog editor automatically determines the scroll range so that it encloses all dialog elements. If the dialog's scroll bars are enabled, they are displayed if the dialog's size is smaller than the scroll range.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 - 9999

## SELECTED

Determines whether a particular item in a list box control is selected.

<b>Applies to</b>	List box item.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## SELECTED-SUCCESSOR

This attribute helps you determine which items have been selected inside a list box control or a selection box control. The first selected item will be retrieved by the control itself, the next selected item will be retrieved as the handle of the list box item or selection-box item. A handle value other than NULL-HANDLE is returned by selected items, but can also be returned by non-selected items. For programming techniques related to the SELECTED-SUCCESSOR attribute, please refer to **Working with List Box Controls and Selection Box Controls** in the section Event-driven Programming Techniques of your Natural User's Guide.

<b>Applies to</b>	List box control, list box item, selection box control.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any dialog element handle

## SERVER-OBJECT

Determines the file name of an external OLE object associated with an OLE server application. If you assign this value, the corresponding object is loaded into the OLE container control. When this object is loaded, the current value of the attributes SERVER-PROGID and EMBEDDED-OBJECT is removed.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	A253
<b>Default Value</b>	None
<b>Possible Values</b>	Any file name

## SERVER-PROGID

Determines the programmatic identifier of an OLE server application. If you assign this value, a specific OLE server application is selected for the OLE container control. Any object previously loaded into the OLE container control is removed. The current values of the attributes SERVER-PROGID and EMBEDDED-OBJECT are also removed.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	A253
<b>Default Value</b>	None
<b>Possible Values</b>	Any program ID

## SHARED

Determines the MDI dialog which is to receive the event notifications for a dialog element. If this attribute is set to FALSE, the events are sent to the owning dialog (typically the MDI frame dialog). If this attribute is set to TRUE, the events are sent to the active MDI child dialog (if any), or to the MDI frame dialog if no MDI child dialog is active. For non-MDI dialogs, this attribute has no effect.

<b>Applies to</b>	Menu item, signal, status bar pane, tool bar item.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## SIZE-MODIFIABLE

Determines whether the dialog's size is modifiable by the end user or not. This implies that the dialog is surrounded by a thick border.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## SLIDER

Determines the position of the slider in a scroll bar control inside the numeric range limited by the values of the MIN and MAX attributes.

<b>Applies to</b>	Scrollbar control.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	All integer values between the MIN and the MAX value of the scroll bar control.

## SORTED

Determines whether the items in a list box or in a selection-box are sorted when inserted into the box. Note that if you have SORTED list box item STRINGS and selection-box item STRINGS, you cannot modify them.

<b>Applies to</b>	List box control, selection box control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	FALSE
<b>Possible Values</b>	TRUE / FALSE

## STATUS-HANDLE

Specifies the status bar control (if any) which is to receive output in the case where no status bar control handle is explicitly specified (for example, when the STATUS-TEXT attribute of the dialog is changed). If the value of this attribute is NULL-HANDLE, the output will be sent to the implicit status bar (if any) that is created when the dialog's HAS-STATUS-BAR attribute is set to TRUE.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / status bar control handle

## STATUS-TEXT

By modifying the value of this attribute, you display the assigned text in the status bar of a dialog. The text will only be displayed if HAS-STATUS-BAR is TRUE. Then the DIL-TEXT will overlap the STATUS-TEXT value or vice versa, depending on which was modified last. For MDI child windows, the status bar is located in the MDI frame window.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string / any text string

## STRING

Determines the text string associated with a dialog element, for example, the title of a dialog window or the label of a push button control. For a font control, this text string is invisible to the end user.

By default, the system font is used to display the STRING. You should, however, be aware that the size of the system font depends on your operating system and on your screen driver. As a result of this, a STRING may not fit in the dialog element's rectangle.

<b>Applies to</b>	Column specification control, dialog (window, MDI frame, MDI child), edit area control, font control, graphic text control, group frame control, input field control, list box item, menu item, push button control, radio button control, selection box control, selection-box item, status bar control, status bar pane, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string / any text string

## STYLE

Determines a set of properties of the above-mentioned dialog elements. For one instance of a dialog element, several properties may be set. The style of a dialog element is represented by a set of characters. Each character determines a property specific to the dialog element.

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, column specification control, dialog (window), edit area control, graphic text control, input field control, line control, OLE container control, push button control, rectangle control, scroll bar control, selection box control, status bar control, status bar pane, table control, text constant control, tool bar, tool bar control, tool bar item. The attribute applies only to those ActiveX controls that behave like buttons. These ActiveX controls are marked in the system registry with the style OLEMISC_ACTSLIKEBUTTON.
<b>Data Type</b>	A253

### Possible Values for ActiveX Controls

Value	Meaning
O	OK button: is pushed if the end user presses ENTER. This attribute is available only for ActiveX controls with the style OLEMISC_ACTSLIKEBUTTON.
C	Cancel button: is pushed if the end user presses ESC. This attribute is available only for ActiveX controls with the style OLEMISC_ACTSLIKEBUTTON.

### Possible Values for Bitmap Controls

Value	Meaning
c	Align the bitmap to the center of the dialog element's rectangle.
l	Align to the left.
r	Align to the right.
v	Align to the vertical center.
t	Align to the top.
b	Align to the bottom.
s	Scale the bitmap to fit into the dialog element's rectangle.
F	Three-dimensional frame.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as "default", the color of the first pixel in the bitmap is assumed to be the background color.

### Possible Value for Canvas Controls

Value	Meaning
F	Frame (border) around the canvas control (reduces the visible area of the canvas).

### Possible Values for Control Box Controls

Value	Meaning
F	Framed: Frame around the control box control.
L	Lowered: Creates a 3-D border with a sunken appearance.
E	Exclusive: Hide all other sibling control boxes with this style automatically when this control is made visible.
T	Transparent: (The control itself is not visible, but children may be).
z	Resize: To fill parents client area whenever parent is resized.

#### Possible Values for Dialogs (WINDOW)

Value	Meaning
c	Center dialog on screen.
P	Modeless (Pop-up; window is independent of its parent).
M	Modal.
X	Dialog box.
d	Default position; will be ignored if "Dialog box" is set as well..
D	Default rectangle; will be ignored if "Dialog box" is set as well.
Z	Control clipping. Prevents controls overprinting other controls which are "in front" in the so-called Z order. The position of a control in the Z-order sequence is determined via the control's SUCCESSOR attribute.
3	3-D border: Dialog interior drawn with sunken 3-D appearance.

#### Possible Values for Edit Area Controls

Value	Meaning
w	Wordwrapped (if text exceeds the width of the edit area control, it is automatically wrapped to the next line).
F	Frame around the edit area control.
v	Autoscroll: Text is automatically scrolled upwards if ENTER key is pressed on the last displayed line. This style also enables scrolling in the absence of a vertical scroll bar (e.g. via cursor keys or mouse wheel).Note: An edit area with a vertical scroll bar is implicitly autoscrollable.

#### Possible Value for Group Frame Controls

Value	Meaning
p	Group frame becomes parent of any controls placed within it.

#### Possible Values for Input Field Controls

Value	Meaning
l	Left-justified input.
r	Right-justified input.
c	Horizontally centered input.
N	Non-displayed input (for example, for passwords).
U	Upper-case input.
M	Input is mandatory.

#### Possible Values for Line Controls and Rectangle Controls

Value	Meaning
S	Solid line (default).
-	Dashed line pattern.
.	Dotted line pattern.
!	Dash-dot line pattern.
:	Dash-dot-dot line pattern.

#### Possible Value for List Box Control

Value	Meaning
3	3-D border: Border with sunken 3-D appearance around the list box control.
I	Integral height. Partial rows are not displayed.

#### Possible Values for Menu Items

Value	Meaning
s	Scaled: Bitmaps are scaled to match the image height and width specified for the parent menu.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as "default", the color of the first pixel in the bitmap is assumed to be the background color.

#### Possible Value for OLE Container Controls

Value	Meaning
F	Framed: Frame around the OLE container control.

#### Possible Values for Push Button Controls

Value	Meaning
O	OK button: is pushed if the end user presses ENTER.
C	Cancel button: is pushed if the end user presses ESC.

**Possible Values for Scrollbar Controls**

Value	Meaning
h	Slider will scroll horizontally.
v	Slider will scroll vertically.

**Note:**

When you edit the STYLE attribute value in the scroll bar control attributes window, setting "h" instead of "v" and vice versa, the RECTANGLE-H and RECTANGLE-W attribute values are exchanged. The dialog editor thus ensures that the scroll bar control will not provide for vertical scrolling in a horizontal shape and vice versa.

**Possible Values for Selection Box Controls**

Value	Meaning
M	Input into this selection box control is mandatory.
X	Box is pulled down all the time.

**Possible Values for Status Bar Controls**

Value	Meaning
3	3-D border: border will be drawn with a 3-D appearance.
t	Top border: border will be drawn along top edge of control.
b	Bottom border: border will be drawn along bottom edge of control.
g	Gripper: control has a sizing grip.

**Possible Values for Status Bar Panes**

Value	Meaning
c	Centered: text will be centered within the pane.
H	Hide pane text (instead of graying it out) if pane is disabled.
R	Raised: pane will be drawn with a raised 3-D appearance.
n	No borders: pane will be drawn without borders.

**Possible Value for Submenus and Context Menus**

Value	Meaning
c	Cool menu: menu items are drawn with their bitmap (if any).

**Possible Values for Table Controls**

Value	Meaning
h	Columns header: buttons with field names are displayed at the top of each column.
I	Integral height: partial rows are not displayed.
n	No lines: the table control is displayed without the lines that normally separate the cells.
s	Single cell selection: If this attribute is set, end users may only select single cells. If not set, end users may select ranges of cells.
w	Whole row selection: selecting an individual cell sets the selection to the entire row.
e	Extendable: end users can delete and insert rows using DEL and INS.
c	Resize columns: end users may resize the columns horizontally.
r	Resize rows: end users may resize the rows vertically.
d	Draggable columns: If this attribute is set, end users may drag the columns.

#### Possible Values for Text Constant Controls

Value	Meaning
l	Left-justified input.
r	Right-justified input.
c	Horizontally centered input.
F	Frame around the text constant control.

#### Possible Values for Tool Bars

Value	Meaning
w	Wrapped: if set and there are more tool bar items than can be displayed on the top of the dialog, the tool bar wraps around to a new line. (The default: the tool bar can be scrolled with the two small arrow push buttons on the left of the tool bar.)

#### Possible Values for Tool Bar Controls

Value	Meaning
3	3-D border: border will be drawn with a 3-D appearance.
t	Top border: border will be drawn along top edge of control. This style is implicitly set for dockable tool bar controls.
b	Bottom border: border will be drawn along bottom edge of control. This style is implicitly set for dockable tool bar controls.
l	Left border: border will be drawn along left edge of control. This style is implicitly set for dockable tool bar controls.
r	Right border: border will be drawn along right edge of control. This style is implicitly set for dockable tool bar controls.
g	Gripper: control has a gripper bar.
L	Tool bar items are drawn with a flat (rather than raised) appearance.
Y	Dynamic: tool bar control cannot be resized when floated, and cannot contain any child controls.

**Possible Values for Tool Bar Items**

<b>Value</b>	<b>Meaning</b>
s	Scaled: bitmaps are scaled to fit on the tool bar items.
T	Transparent: Bitmap pixels matching the background color are not drawn. If the background color is specified as "default", the color of the first pixel in the bitmap is assumed to be the background color.

## SUCCESSOR

Determines the next sibling of a dialog element in the parent/child hierarchy. A SUCCESSOR refers to the next created dialog element on the same level (with the same PARENT). You can use this attribute to go through all dialog elements in one parent dialog or dialog element.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / handle value of the successor

## SUPPRESS-ACTIVATE-EVENT

Determines whether the activate event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the corresponding OLE server application has just been activated in-place.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-BEFORE-OPEN-EVENT

Determines whether the before-open event will be suppressed for the context menu (or submenu) or not. If it is suppressed, the event handler will not get control when a context menu (or submenu) is displayed. Note that this attribute cannot be used to suppress the before-open event for a dialog.

<b>Applies to</b>	Context menus, Submenus
<b>Data Type</b>	I4
<b>Default Value</b>	1 (SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-CHANGE-EVENT

Determines whether the change event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the content of an input field control changes, for example.

<b>Applies to</b>	Canvas control, column specification control, edit area control, input field control, OLE container control, scroll bar control, selection box control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-CLICK-EVENT

Determines whether the click event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user makes a selection.

<b>Applies to</b>	Bitmap control, canvas control, column specification control, list box control, OLE container control, radio button control, status bar pane, table control, timer, toggle button control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-CLIENT-SIZE-EVENT

Determines whether the client-size event will be suppressed for the dialog or not. If it is suppressed, the event handler will not get control when the dialog's interior size changes. The dialog interior is the part of the dialog which does not include the menu bar (if any), tool bars (if any) or status bar (if any).

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	1 (SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

## SUPPRESS-CLOSE-EVENT

Determines whether the close event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the corresponding OLE server application is about to be terminated.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-DBL-CLICK-EVENT

Determines whether the double-click event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user double-clicks an item in a list box control.

<b>Applies to</b>	Bitmap control, canvas control, column specification control, list box control, OLE container control, status bar pane, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-DELETE-ROW-EVENT

Determines whether the delete-row event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user deletes a row.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-ENTER-EVENT

Determines whether the enter event will be suppressed for the dialog (element) or not. If it is suppressed, the event handler will not get control when the dialog (element) gets the focus or when the dialog is activated.

<b>Applies to</b>	Column specification control, dialog (window, MDI frame, MDI child), edit area control, input field control, selection box control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-ENTER-CELL-EVENT

Determines whether the enter-cell event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user enters a cell.

<b>Applies to</b>	Column specification control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-FILL-EVENT

Determines whether the fill event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user scrolls to the beginning or the end of the control (for example, the list box control).

<b>Applies to</b>	List box control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-IDLE-EVENT

Determines whether the idle event will be suppressed for the dialog or not. If it is suppressed, the event handler will not get control when an action occurs which could affect the state of the dialog's user interface (for example, when a key or mouse button is pressed or released).

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	1 (SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

## SUPPRESS-INSERT-ROW-EVENT

Determines whether the insert-row event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user is about to insert a row.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-LEAVE-EVENT

Determines whether the leave event will be suppressed for the dialog (element) or not. If it is suppressed, the event handler will not get control when the dialog (element) loses the focus or when a dialog is deactivated.

<b>Applies to</b>	Column specification control, dialog (window, MDI frame, MDI child), edit area control, input field control, selection box control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-LEAVE-CELL-EVENT

Determines whether the leave-cell event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user leaves a cell.

<b>Applies to</b>	Column specification control, table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-SIZE-EVENT

Determines whether the size event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user changes the dialog's size.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## SUPPRESS-COMMAND-STATUS-EVENT

Determines whether the command-status event will be suppressed for the dialog or not. If it is suppressed, the event handler will not get control when the state of the dialog's command UI (for example, menu items and tool bar items) is to be updated during idle processing.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	I4
<b>Default Value</b>	1 (SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

## SUPPRESS-TOP-EVENT

Determines whether the top event will be suppressed for the dialog element or not. If it is suppressed, the event handler will not get control when the end user navigates to the top of a table control.

<b>Applies to</b>	Table control.
<b>Data Type</b>	I4
<b>Default Value</b>	0 (NOT SUPPRESSED)
<b>Possible Values</b>	0 (NOT SUPPRESSED) / 1 (SUPPRESSED)

**Note:**

You can use the symbols SUPPRESSED or NOT-SUPPRESSED defined in the local data area NGULKEY1. NGULKEY1 is automatically included in your application.

## TIMER-INTERVAL

Determines the time interval (in milliseconds) at which the click event handler of the timer will be triggered, that is, the click event handler will be triggered every  $n$  milliseconds.

<b>Applies to</b>	Timer.
<b>Data Type</b>	I4
<b>Default Value</b>	0
<b>Possible Values</b>	0 or any other integer value

## TOOLBAR-HANDLE

Associates a tool bar and a dialog. Several dialogs may share one tool bar. Please note that the handle value of the tool bar must be assigned to the TOOLBAR-HANDLE attribute before you can make the tool bar visible with the HAS-TOOLBAR attribute.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	HANDLE
<b>Default Value</b>	NULL-HANDLE
<b>Possible Values</b>	NULL-HANDLE / any handle of a tool bar

## TOOLBAR-POS

Specifies the position of the tool bar relative to the associated dialog (top, bottom, left, right).

<b>Applies to</b>	Dialog (window, MDI Frame).
<b>Data Type</b>	I4
<b>Default Value</b>	TB-TOP
<b>Possible Values</b>	TB-TOP / TB-BOTTOM / TB-LEFT / TB-RIGHT

## TOOLTIP

Specifies a dialog element's tool tip text, which is automatically displayed in a small pop-up window when the mouse pointer hovers over the dialog element. For menu items, the tool tip text is not displayed directly, but is "inherited" by any tool bar items referencing this menu item via their SAME-AS attribute. The specified tool tip text is only displayed if it is not an empty string and if the parent control's HAS-TOOLTIP attribute is set to TRUE.

<b>Applies to</b>	Menu item, signal, status bar pane, tool bar item.
<b>Data Type</b>	A253
<b>Default Value</b>	Empty string
<b>Possible Values</b>	Empty string / any text string

## TYPE

Determines the type of dialog element to be created next. You can also use this attribute to query the type of an existing dialog element. You must use the TYPE value as a parameter in the PROCESS GUI statement action ADD. This will determine which type of dialog element is to be created.

<b>Applies to</b>	Dialogs (window, MDI frame, MDI child) and all dialog elements.
<b>Data Type</b>	I4
<b>Default Value</b>	undefined

### Possible Values

BITMAP (0)	CANVAS (24)
COLUMNSPECIFICATION (29)	EDITAREA (1)
CONTEXTMENU (32)	CONTROLBOX (33)
FONT (2)	GRAPHICTEXT (27)
GROUPFRAME (3)	INPUTFIELD (4)
LINE (25)	LISTBOX (5)
LISTBOXITEM (6)	MENUBAR (7)
MENUITEM (8)	OLECONTAINER (30)
PUSHBUTTON (9)	RADIOBUTTON (10)
RECTANGLE (26)	SCROLLBAR (23)
SELECTIONBOX (11)	SELECTIONBOXITEM (12)
SUBMENU (20)	TABLE (28)
TEXTCONSTANT (13)	TIMER (22)
TOGGLEBUTTON (14)	TOOLBAR (15)
TOOLBARITEM (16)	WINDOW (17)
MDIFRAME (18)	MDICHILD (19)

**Examples of usage in the PROCESS GUI statement action ADD:**

```
PROCESS GUI  
ACTION ADD WITH #LISTBOX LISTBOXITEM #ITEM PROCESS GUI ACTION ADD WITH PARAMETERS  
PARENT = #LISTBOX STRING = 'Test' TYPE = LISTBOXITEM HANDLE-VARIABLE = #ITEM END-PARAMETERS
```

**Note:**

The text representation can be substituted for the integer representation because the local data area NGULKEY1 (supplied in library SYSTEM) is automatically included in the event handler code.

## VARIABLE

Links a Natural variable such as EMP.PERSONNEL-ID to the dialog element. This attribute must always be used with the LINKED attribute. As an alternative to using the LINKED and the VARIABLE attributes, you can also specify "Linked Variable" in a "Source" dialog box of an attributes window.

<b>Applies to</b>	Input field control, selection box control.
<b>Data Type</b>	Any Natural data type
<b>Default Value</b>	No variable associated
<b>Possible Values</b>	Any Natural variable

## VERSION

Indicates the Natural version with which the dialog was saved last; you can query it.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child).
<b>Data Type</b>	A253
<b>Default Value</b>	223
<b>Possible Values</b>	Any Natural version (plus patch level information, if applicable).

## VERT-SCROLLABLE

Determines whether the dialog or edit area control has a vertical scroll bar.

<b>Applies to</b>	Dialog (window, MDI frame, MDI child), edit area control, table control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

## VISIBLE

If this attribute is set, the dialog element is visible. You can use this attribute to hide dialog elements by setting it to FALSE, if, for example, a logical condition is fulfilled and displaying the dialog or dialog element does not make sense under this conditio

<b>Applies to</b>	ActiveX control, Bitmap control, canvas control, column specification control, control box control dialog (window, MDI frame, MDI child), edit area control, graphic text control, group frame control, input field control, list box control, line control, menu bar, OLE container control, push button control, radio button control, rectangle control, scroll bar control, selection box control, signal, submenu control, status bar control, status bar pane, table control, text constant control, tool bar control, toggle button control.
<b>Data Type</b>	BOOLEAN
<b>Default Value</b>	TRUE
<b>Possible Values</b>	TRUE / FALSE

n.

## ZOOM-FACTOR

Magnifies or reduces the default representation of an OLE server object that has become visible in an OLE container control.

If you set this attribute to the value "0", the server application's object is scaled so that it fits into the OLE container control's rectangle. To avoid a distorted display, the OLE container control's rectangle should have a width/height ratio similar to that of the server object.

<b>Applies to</b>	OLE container control.
<b>Data Type</b>	I4
<b>Default Value</b>	100
<b>Possible Values</b>	0 to 500

# Events

---

## Activate Event

### Applies To

OLE container control.

### Description

This event handler section is performed when the corresponding OLE server application is about to be activated in place.

## After-Any Event

### Applies To

Dialog.

### Description

This event handler section is performed after each event in an application and is therefore specified only once. You use it, for example, to modify variables uniformly across the application, or to perform checks. When using this type of event, make sure this does not interrupt the processing of the events that are performed afterwards.

You can use the system variables \*CONTROL and \*EVENT to determine which dialog element has received which event.

#### Note:

The after-any event will not occur after a close event.

## After-Open Event

### Applies To

Dialog.

### Description

This event handler section is performed after the dialog with all child dialog elements has been created and displayed. You can use it, for example, to create dialog elements dynamically by using the ADD action of the PROCESS GUI statement in its event handler code. You can also use it to add items to list-box controls or selection-box controls with the corresponding PROCESS GUI statement action.

## Before-Any Event

### Applies To

Dialog.

### Description

This event handler section is performed before each event in an application and is therefore specified only once. You use it, for example, to modify variables uniformly across the application, or to perform checks. When using this type of event, make sure this does not interrupt the processing of the event handlers that are performed afterwards.

## Before-Open Event

### Applies To

Dialog, context menu, submenu control.

### Description

Occurs before the dialog is created and displayed. You can use it, for example, to load profile information from a database. For context menus and submenus, the event occurs immediately before the menu is displayed, allowing menu items to be checked, disabled, etc., according to the current context.

#### **Note:**

As the dialog elements are created after this dialog event, all handle variables except #DLG\$PARENT are NULL. Note also that in the case of dialogs, this event cannot be suppressed, although for context menus and submenus it can be.

## Change Event

### Applies To

Edit area control, input-field control, OLE container control, scroll bar control, selection-box control, table control.

### Description

Occurs whenever the `STRING` attribute value of a dialog element has changed, for example, as a result of keyboard input or as a result of a dynamic change at runtime. Occurs for a scroll bar control when the `SLIDER` attribute value changes. Occurs for an OLE container control when the object's data have been changed. The event handler associated with this event would typically be used to enable other dialog elements when an input-field was filled (such as in an input-field control or in a selection-box control).

If this event occurs in a cell of a table control, the `ROW` and `COLUMN` attributes are modified so that they contain the position of the changed cell.

If this event occurs for an input-field control or a selection-box control with a linked variable, the linked variable must be updated in the change event.

## Click Event

### Applies To

Bitmap control, list-box control, menu item, OLE container control, push button control, radio-button control, table control, toggle-button control, tool-bar item.

### Description

Occurs whenever the end user selects the dialog element by a keyboard function or by a mouse click: a push-button control may be pressed, a list box item may be selected, or a toggle-button control may be checked. It is the most important event because the end user may trigger business logic with it (press a push-button control to trigger a calculation or to validate data that have been entered, and so on).

Clicking a dialog element always selects the dialog element, but it does not necessarily cause an action to occur. A click event handler section is therefore only relevant if you want an action to occur.

If this event occurs in a cell of a table control, the `ROW` and `COLUMN` attributes are modified so that they contain the position of the cell that has been clicked on.

## Client-Size Event

### Applies To

Dialog.

### Description

Occurs whenever the interior size of the dialog window changes, for example, if a tool-bar control is re-docked at a different location. The dialog interior is the part of the dialog which does not include the menu bar (if any), tool bars (if any) or status-bar (if any). You can use the client-size event handler to size controls in the dialog window relative to the dialog window's size.

## Close Event

### Applies To

Dialog, OLE container control.

### Description

Occurs whenever the end user selects the Close option on the Natural menu to close a dialog window or whenever the dialog is closed using the CLOSE DIALOG statement or whenever an OLE container application is closed. It can be used to make the associated event handler check if all work has been saved and ask the user to save and close, exit and close, or cancel. When a dialog is closed, all child dialogs are also closed. In this case, the close event handlers of the child dialogs are not triggered. If you want to close the parent dialog and trigger the close event handlers in the child dialogs as well, you use the subprogram NGU-DIALOG-CLOSE-ALL.

## Command-Status Event

### Applies To

Dialog.

### Description

This event occurs whenever one or more commands (signals, menu items or tool bar items without a SAME-AS attribute, or status bar panes) need to be updated (i.e., disabled, checked, etc.) by a dialog during idle processing or in response to an explicit UPDATE-COMMAND-STATUS action. The commands do not necessarily belong to the dialog receiving the command-status event if their SHARED attribute is set to TRUE, which causes MDI frame commands to be automatically redirected to the active MDI child dialog (if any). This is precisely the case where the command-status event is most useful, because it allows multiple instances of an MDI child dialog (or even completely different MDI child dialogs) to share the same tool bar and status bar controls without interfering with each other, as would be the case if each MDI child would attempt to update the commands directly.

The intended usage of the command-status event is to only update state variables (and not the commands themselves) when the program state changes, and then to do a bulk update of the commands according to the current values of the state variables within the command-status event handler.

Note that the command-status event handler is called frequently, and therefore should return as quickly as possible. For this reason, database access (for example) should be avoided in this event.

If there are no commands which need to be serviced by a particular dialog, no command-status event is raised, even if there is code available for it and the event is not suppressed. This is the case for an active MDI child if no commands have been marked as shared, for example.

For performance reasons, it is not possible for the application to find out which commands caused a particular command-status event to be triggered. Instead, a dialog should update all commands for which it is the current target, and which require a non-default state to be set. The default state is disabled and unchecked for signals, menu items and tool bar items and invisible for status bar panes. Any commands not explicitly enabled or disabled (or any status bar panes not explicitly shown or hidden) by the program in the command-status event will be automatically reset to the default state by the system. This allows a particular MDI child dialog, for example, to only set the status of commands it "knows" about, and to let the system implicitly reset the commands intended only for different child dialogs. Thus, if a command is introduced for a specific MDI child dialog only, the command-status events of the all other MDI child dialogs do not need to be modified. This automatic command resetting is not performed if the corresponding dialog's command-status event is

suppressed.

The command-status event for a particular dialog is raised before the idle event (if any) for that dialog, in order that the effects of the idle event code can be taken into account by the command status updating process.

## DDE-Client Event

### Applies To

Dialog.

### Description

Occurs when your dialog is acting as a client of a DDE conversation and receives a DDE message from the DDE server application, for example Microsoft Excel. The field DDE-VIEW.MESSAGE (or its equivalent) contains one of the following values:

- DISCONNECT
- DATA
- NOTIFY
- TIMEOUT

## DDE-Server Event

### Applies To

Dialog.

### Description

Occurs when your dialog is acting as a client of a DDE conversation and receives a DDE message from a DDE client application, for example Microsoft Excel. The field DDE-VIEW.MESSAGE (or its equivalent) contains one of the following values:

- CONNECT
- DISCONNECT
- REQUEST
- ADVISE
- UNADVISE
- POKE
- EXECUTE
- TIMEOUT

## Default Event

### Applies To

Any dynamically created dialog element.

### Description

Occurs whenever a non-suppressed event occurs for which no event handler section is specified. Also occurs whenever the end user triggers an event in a dynamically created dialog element. The default event handler section can be used to DECIDE which event (value of \*EVENT) occurred for which dialog element (value of \*CONTROL). For each particular dialog element and event, you can then specify the individual event handler code section.

A default event occurs, for example, if you issue SEND EVENT 'HUGO', but no such 'HUGO' event has been defined. Another example is that the event is not suppressed, but the event handler section is empty.

## Delete-Row Event

### Applies To

Table control.

### Description

Occurs after the end user has deleted a row by selecting it and by pressing DEL or by calling "PROCESS GUI ACTION TABLE-DELETE-ROW ...". The table control's STYLE attribute must have had the value "e" (extendable, allows end users to delete rows).

If this event occurs in a cell, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

## Double-Click Event

### Applies To

Bitmap control, canvas control, list-box control, OLE container control, table control.

### Description

Occurs whenever the end user double-clicks on the dialog element. For list-box controls and radio-button controls, the double-click event normally triggers the event handler with the business logic, whereas the click event would normally trigger an event handler with a simple display function.

The first click of a double-click triggers a click event, except for when the click event is suppressed. In the context of list-box controls and radio-button controls, the first click of the double-click selects a list box item or a radio-button control. The second click should be equivalent to pressing the default button in the corresponding window.

If this event occurs in a cell of a table control, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

If this event occurs inside the rectangle of an OLE container control, the OLE server application is started. Double-clicking inside an OLE container control is equivalent to clicking with the right mouse button. (Clicking the right mouse button inside the OLE container control's rectangle invokes a pop-up menu. By default, releasing the mouse button without further selection executes the first command verb of the OLE server application.)

## Drag-Drop Event

### Applies To

Bitmap control.

### Description

Whenever the end user points to a bitmap control with the mouse, holds the mouse button down, drags it onto a target bitmap control, this event occurs for the target bitmap control on releasing the mouse button (dropping the dragged bitmap control). In the event handler code of this event, you can use the PROCESS GUI statement action INQ-DRAG-DROP to find out where the bitmap control was dropped.

## Enter Event

### Applies To

Dialog, edit area control, input-field control, selection-box control, table control.

### Description

Occurs whenever the end user changes the focus by clicking on the dialog element or by using TAB.

For dialogs, this event occurs when a dialog becomes active. This happens when

- the end user activates it by a mouse click or by a keyboard operation; or
- the PROCESS GUI statement action SET-FOCUS is applied to this dialog.

For dialog elements, this event occurs when the dialog element receives the focus by end user action or if the SET-FOCUS action is applied.

For table controls, this event occurs if the table receives the focus or if the end user enters a new *row*. If the end user enters a new *cell*, the enter-cell event occurs.

If this event occurs in a cell of a table control, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

When an end user leaves an MDI application, for example, by clicking into another application outside Natural, and clicks into the MDI application again, an enter event only occurs for the MDI Frame dialog, not for the MDI Child dialogs.

## Enter-Cell Event

### Applies To

Table control.

### Description

Occurs whenever the end user changes the focus by clicking on a cell in a table control, by using TAB to enter a cell, or by using the arrow keys to enter a cell.

If this event occurs, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

## Error Event

### Applies To

Dialog.

### Description

This event handler section is performed whenever a runtime error occurs while a dialog is active. You can specify event handler code to be executed whenever this error occurs. If no error event handler code is specified, Natural will terminate with an error message and all dialogs will be closed. If an error event handler code section was specified, the dialog will remain open and handle the error.

You can use the system variables \*CONTROL and \*EVENT to determine the event handler in which the error occurred. If you want the dialog to continue after processing the error, code an ESCAPE ROUTINE statement at the end of the error event handler.

## Fill Event

### Applies To

List-box control, table control.

### Description

Occurs for list-box controls whenever the end user scrolls to the end of a list box control that has scroll bars. Occurs for table controls whenever the end user navigates to the last row in the table control or scrolls to the end of the table control with the scroll bar. If all items in the list-box control are visible without scrolling, there is no scroll bar and therefore no fill event will occur. In this event, you would usually append additional items to the end of the list-box control with the PROCESS GUI statement action ADD-ITEMS.

If this event occurs in a cell of a table control, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

## Idle Event

### Applies To

Dialog.

### Description

Occurs during idle-time processing in situations where the user interface may need to be updated, which is after each time a key or mouse button is pressed or released. Note that mere mouse movement does not normally cause idle events to be raised, because this would be too slow. An exception is when the mouse pointer is moved away from a tool bar, in order to allow the program to update the status-bar text.

In addition to allowing the program to update the status-bar with an idle message (e.g., "Ready"), the idle event may also be used for monitoring user interface changes. For example, the idle event of an MDI frame dialog can query the active MDI child dialog and show or hide individual tool bars appropriately if necessary.

Note that the idle event handler is called frequently, and therefore should return as quickly as possible. For this reason, database access (for example) should be avoided in this event.

## Insert-Row Event

### Applies To

Table control.

### Description

Occurs after the end user has inserted a row by pressing INS or by calling "PROCESS GUI ACTION TABLE-INSERT-ROW ...". The table control's STYLE attribute must have had the value "e" (extendable, allows end users to insert rows).

If this event occurs in a cell, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

# Leave Event

## Applies To

Dialog, edit area control, input-field control, selection-box control, table control.

## Description

Occurs whenever the dialog or dialog element loses the focus. You can use the leave event handler to validate field entries or to dynamically modify other dialog elements whose status depends on that of the dialog element just left. Does not occur on closing a dialog.

For table controls, this event occurs if the end user leaves a *row* or if the table loses the focus. If the end user leaves a *cell*, the leave-cell event occurs.

If this event occurs in a cell of a table control, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

Before this event handler code is processed, end user input in (linked variable) input-field controls and selection-box controls is checked against any EDIT-MASK value or applicable Natural data type. However, in the following situations the leave event is *not* triggered after end user input was checked and a linked variable was updated:

- An input-field control or a selection-box control has the focus and the end user clicks a menu item; or
- the end user presses ENTER, triggering the default button; or
- the end user presses ENTER, triggering the OK button; or
- the end user presses an accelerator key.

### Note:

It is not recommended to execute the PROCESS GUI statement action SET-FOCUS from the leave event because the leave event implies that the focus is to be set to yet another part of the application.

## Leave-Cell Event

### Applies To

Table control.

### Description

Occurs whenever the end user moves the focus away from a cell (for example, by clicking on another cell in the table control or by using TAB to enter another cell).

If this event occurs in a cell, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

## Size Event

### Applies To

Dialog.

### Description

Occurs whenever the size of the dialog window changes, for example, if the window is minimized or maximized. You can use the size event handler to size controls in the dialog window relative to the dialog window's size.

## Top Event

### Applies To

Table control.

### Description

Occurs whenever the end user navigates to the top of the table control (either by using the scroll bar or by entering a cell of the top row).

If this event occurs in a cell, the ROW and COLUMN attributes are modified so that they contain the position of the cell.

## User-Defined Events

### Apply To

Dialog.

### Description

User-defined events occur whenever you have specified a SEND EVENT statement with the name of a previously undefined event in an event handler code of a dialog. Note that the user event must be defined in another dialog than the one where it occurs. For more details and examples of how to cause an action to occur in another dialog, see the section Event-driven Programming Techniques of your Natural User's Guide for Windows.

You can define user events for dialogs by pressing the "New" button in the "Events..." dialog box of the dialog's attributes window. You can then enter any name of your newly-defined user event and specify the corresponding event section. It is recommend that you use "#" as the first letter to distinguish user-defined events from previously defined events.

If no event handler is defined for an event generated by the SEND EVENT statement, the default event handler is executed with the value of the system variable \*EVENT as the user event name.

# PROCESS GUI Statement Actions

---

## General Information

The PROCESS GUI statement actions execute procedures from within the PROCESS GUI statement. They are flexible in parameter handling: it is sufficient to supply type-compatible parameters with them (for example, if A5 is the exact parameter type, supplying an I1 parameter will be accepted).

For more information about the PROCESS GUI statement, refer to **Executing Standardized Procedures** in the section Event-Driven Programming Techniques of your Natural User's Guide.

To use the PROCESS GUI statement actions more comfortably, the local data area NGULKEY1 is automatically included in the list of local data areas used by any new dialog.

NGULKEY1 contains reserved symbols to be used in any event handler code. This enables you to use meaningful names as parameters in a PROCESS GUI statement. It also enables you to refer to certain attribute values by the more meaningful texts rather than by the integer values.

**Note:**

The "Response" parameters in the PROCESS GUI statement actions are optional.

**Note:**

ActiveX controls use the PROCESS GUI statement in a slightly different way. You use the statement to execute the ActiveX control's own methods and to access parameterized properties. For a description on how to use the PROCESS GUI statement for ActiveX controls, see your Natural User's Guide - Working with ActiveX Controls.

## ADD Action

### Description

Creates a single specified dialog element dynamically. See also **Creating/Deleting Dialog Elements Dynamically** in the section Event-Driven Programming Techniques of your Natural User's Guide. This action is most frequently used to add items to a list box control or to a selection-box control or to add column specifications to a table. It can also be used to create all kinds of dialog elements dynamically.

There are two syntax options of this action:

#### **PROCESS GUI ACTION ADD WITH ...**

This option has the parameters as listed below. Other attributes of the newly created dialog element have to be set in the global attributes list before the PROCESS GUI statement.

#### **PROCESS GUI ACTION ADD WITH PARAMETERS ... END-PARAMETERS**

This option accepts a list of attribute assignments, one for each attribute that is to be specified for the newly created dialog element. If you use this option, the global attribute list is not used or affected. For all attributes that are not explicitly specified, the default value is taken.

## Parameters for the ADD WITH option

Name/Data Type	Explanation
HANDLE OF GUI	Input The handle of the parent dialog element.
Type (I4)	Input The type of dialog element to be created.
HANDLE OF GUI	Output The handle of the newly created dialog element.
Response (I4)	Output Natural error (if applicable).

### Example 1 (option 1):

```

DEFINE DATA LOCAL
1 #NEW1 HANDLE OF INPUTFIELD
END-DEFINE
...
#NEW1.STRING:= 'NEW1'
#NEW1.RECTANGLE-X:= 24
#NEW1.RECTANGLE-Y:= 30
#NEW1.RECTANGLE-W:= 176
#NEW1.RECTANGLE-H:= 28
#NEW1.ENABLED:= TRUE
#NEW1.VISIBLE:= TRUE
PROCESS GUI ACTION ADD WITH #DLG$WINDOW INPUTFIELD #NEW1

```

### Example 2 (option 2):

```

DEFINE DATA LOCAL
1 #NEW2 HANDLE OF INPUTFIELD
END-DEFINE
...
PROCESS GUI ACTION ADD WITH PARAMETERS
HANDLE-VARIABLE = #NEW2
TYPE = INPUTFIELD
STRING = 'NEW2'
RECTANGLE-X = 24
RECTANGLE-Y = 30
RECTANGLE-W = 176
RECTANGLE-H = 28
ENABLED = TRUE
VISIBLE = TRUE
PARENT = #DLG$WINDOW
END-PARAMETERS

```

If you insert a new dialog element dynamically by using the ADD action, you determine its position in the navigation sequence by creating the dialog element and setting the SUCCESSOR attribute to the handle value of its successor.

### Example:

```
/* Insert input-field control #NEW1 before push button control #PB-1
/* Be careful not to trigger the PROCESS GUI statement action from a push
/* button control named #PB-1 because you are already defining it
DEFINE DATA LOCAL
1 #NEW1 HANDLE OF INPUTFIELD
1 #PB-1 HANDLE OF PUSHBUTTON
END-DEFINE
...
PROCESS GUI ACTION ADD WITH PARAMETERS
  PARENT = #DLG$WINDOW
  HANDLE-VARIABLE = #NEW1
  TYPE = INPUTFIELD
  SUCCESSOR = #PB-1
  ...
END-PARAMETERS
```

## ADD-ITEMS Action

### Description

Adds several list box items in a list box control at once. Does the same for selection-box items in a selection-box control and column specifications in a table control.

### Parameters

Name/Data Type	Explanation
HANDLE OF dialog element	Input Specifies a dialog element.
Number of Items (I4)	Input You can add any number of items.
Item (list of A253-compatible values or one-dimensional array)	Input Item string(s).
Response (I4)	Output Natural error (if applicable).

### Example:

```

DEFINE DATA LOCAL
1 #AMOUNT (I4)
1 #ITEM (A20/1:5)
1 #RESPONSE (I4)
END-DEFINE
...
#AMOUNT:= 5
#ITEM(1):= 'Berlin'
#ITEM(2):= 'Paris'
#ITEM(3):= 'London'
#ITEM(4):= 'Milan'
#ITEM(5):= 'Madrid'
PROCESS GUI ACTION ADD-ITEMS WITH #LB-1 #AMOUNT #ITEM(*) GIVING #RESPONSE

```

## ADD-ITEMS-EX Action

### Description

Adds list box items in a list box control and selection-box items in a selection-box control. Associates data to the items either as CLIENT-KEY/CLIENT-VALUE pairs or as CLIENT-DATA values. The item strings and the corresponding data can be specified either as single values or as one-dimensional arrays. Both strings and data must be provided in the same way. If the parameter "Client key" is blank, each data value is entered as the CLIENT-DATA of the corresponding item. If it is not blank, it is entered as the CLIENT-VALUE with the corresponding CLIENT-KEY.

### Parameters

Name/Data Type	Explanation
HANDLE OF dialog element	Input Specifies a dialog element.
Number of Items (I4)	Input
Client key (A253)	Input CLIENT-KEY attribute to be used for each value in the list of values.
String list (A253)	Input A number of alphanumeric variables or constants or an array specification. These strings are entered in the STRING attribute of each item. You may only use a one-dimensional array in the array specification. Using part of a higher-level array causes an error.
Value list (A253 or I4)	Input A number of alphanumeric variables or constants or an array specification. These are interpreted as the CLIENT-VALUE of the given CLIENT-KEY. If the CLIENT-KEY value is blank, the value list is entered in the CLIENT-DATA.
Response (I4)	Output Natural error (if applicable).

**Example:**

```
/* Definitions in the dialog's local data area:
1 #NUMBER (N4)
1 #CITY (A20/1:2)
1 #CODE (I4/1:2)
1 #KEY (A20)
...
/* Event handler code:
#NUMBER:= 2
#CITY(1):= 'Berlin'
#CODE(1):= 1015
#CITY(2):= 'Munich'
#CODE(2):= 8053
#KEY:= 'Code'
PROCESS GUI ACTION ADD-ITEMS-EX
  WITH #LB-1 #NUMBER #KEY #CITY(1) #CITY(2) #CODE(1) #CODE(2) GIVING #RESPONSE
/* Another possible WITH clause (same result)
  WITH #LB-1 #NUMBER #KEY #CITY(1:2) #CODE(1:2) GIVING #RESPONSE
```

## BEEP Action

### Description

Issues a "beep" sound.

### Parameters

Name/Data Type	Explanation
Number (1 - 3)	Input Type of the beep to be issued. "1" denotes a warning beep, "2" denotes a note beep, "3" denotes an error beep.

### Note:

The "Number" parameter is optional. If you omit this parameter, the value "1" (warning beep) applies by default.

### Example:

```
PROCESS GUI ACTION BEEP
```

## CLEAR Action

### Description

Clears the contents of an input-field control, an edit area control or a table control.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an input-field control, an edit area control or a table control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION CLEAR WITH #EA-1 GIVING #RESPONSE
```

## DELETE Action

### Description

Deletes a specified dialog element. You can use this action, for example, to delete items from a list box control or to remove dialog elements from a dialog.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog element.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION DELETE WITH #PB-1 GIVING #RESPONSE
```

## DELETE-CHILDREN Action

### Description

Deletes all children of a given dialog or dialog element dynamically. You use this action, for example, to delete all items in a list box control before filling the list box again (using the ADD-ITEMS action).

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element whose children are to be deleted.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION DELETE-CHILDREN WITH #LB-1 GIVING #RESPONSE
```

## DELETE-WINDOW Action

### Description

Deletes (unloads) a dialog regardless of whether it has actually been created or not. You use this action, for example, to ensure that the dialog is unloaded if a CLOSE DIALOG statement is performed in the before-open event handler. Normally, you cannot close a dialog before it is actually created. To avoid a conflict, you use this action.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the window which is to be deleted.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION DELETE-WINDOW WITH #MYDIA GIVING #RESPONSE
```

## EDIT-GET-LINE-NUMBER Action

### Description

Retrieves the number of lines in the edit area control. This is also true if the STYLE attribute value has been set to value "w" (wordwrap).

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line number (I4)	Output The number of lines.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION
EDIT-GET-LINE-NUMBER WITH #EA-1
#LINE-NUMBER GIVING
#RESPONSE
#IF-1.STRING:= #LINE-NUMBER
```

## EDIT-LINE-DELETE Action

### Description

Deletes a line in the edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line number (I4)	Output The line to be deleted. If you specify "0", the last line will be deleted.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION EDIT-LINE-DELETE WITH #EA-1  
#LINE-NUMBER GIVING #RESPONSE
```

## EDIT-LINE-GET-SELECTION Action

### Description

Retrieves the extent of the selected area (lines and columns) in an edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line from (I4)	Output Selection starts from this line onwards.
Column from (I4)	Output Selection starts from this column onwards.
Line to (I4)	Output Last selected line.
Column to (I4)	Output The column position immediately following the last selected character.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION
EDIT-LINE-GET-SELECTION WITH #EA-1
#LINE-FROM #COLUMN-FROM
#LINE-TO #COLUMN-TO GIVING #RESPONSE
```

## EDIT-LINE-GET-TEXT Action

### Description

Retrieves text in a line of the edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line number (I4)	Input The text is retrieved out of this line.
Column from (I4)	Input The text is retrieved from this column onwards.
Column to (I4)	Input/Output The text is retrieved up to this column. If you specify "0", the complete line is retrieved. The position immediately following the last character in the line will be returned (or 1, if the line is empty).
Line text (A253)	Output Returns the retrieved text string.
Split (L)	Output Indicates whether the total length of the requested text exceeds the actually retrieved line text string.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#LINE-NUMBER := 1
#COLUMN-FROM := 1
#COLUMN-TO := 0
PROCESS GUI ACTION EDIT-LINE-GET-TEXT WITH #EA-1 #LINE-NUMBER #COLUMN-FROM
#COLUMN-TO #LINE-TEXT #SPLIT GIVING #RESPONSE
```

## EDIT-LINE-INSERT Action

### Description

Inserts a new line into an edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line number (I4)	Input/Output The new line will be inserted before this line. If you specify "0", the new line will be appended to the last line of the edit area control.
Line length (I4)	Input/Output The number of characters to be inserted into the new line starting from the first character of "Line text". If you specify "0", an empty line will be inserted. If you specify "-1", the "Line text" string will be copied into the new line, trailing blanks will be removed and the number of copied characters will be returned.
Line text (A253)	Input/Output The text string of the line to be inserted.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#LINE-NUMBER := 0
#LINE-LENGTH := 10
#LINE-TEXT := 'Hello!'
PROCESS GUI ACTION EDIT-LINE-INSERT WITH #EA-1 #LINE-NUMBER #LINE-LENGTH
#LINE-TEXT GIVING #RESPONSE
```

## EDIT-LINE-SET-SELECTION Action

### Description

Selects a range of text (lines, columns) in an edit area control. To set the text caret to a certain position, "Line from" and "Line to" must have the same value; "Column from" and "Column to" must also have the same value.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line from (I4)	Input/Output Selection starts from this line onwards. If you specify "0", the last line is selected.
Column from (I4)	Input/Output Selection starts from this column onwards. If you specify "0", the end of the line is selected as the starting column.
Line to (I4)	Input/Output Last selected line. If you specify "0", the last line is selected. The number of this last line will be returned.
Column to (I4)	Input/Output Last selected column. If you specify "0", the complete line is selected. The last selected position in the line will be returned.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#LINE-FROM := 1          /* Select the first three lines
#LINE-TO := 3
#COLUMN-FROM := 1       /* Select from the first to the last column
#COLUMN-TO := 0
PROCESS GUI ACTION EDIT-LINE-SET-SELECTION WITH #EA-1 #LINE-FROM #COLUMN-FROM
#LINE-TO #COLUMN-TO GIVING #RESPONSE
```

## EDIT-LINE-SET-TEXT Action

### Description

Replaces text in a line of the edit area control. Note that you can also use this action to

- insert text by setting parameters "Column from" and "Column to" to the same value (this will be the insertion position);  
or to
- delete text by setting the parameter "Line length" to the value '0'.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA	Input Specifies an edit area control.
Line number (I4)	Input/Output The text is replaced out of this line.
Column from (I4)	Input/Output The text is replaced from this column onwards.
Column to (I4)	Input/Output The text is replaced up to this column. If you specify "0", the rest of the line is replaced. The last position in the line will be returned.
Line text (A253)	Input/Output Contains the string replacing the string specified with "Line number", "Column from", "Column to".
Line length (I4)	Input/Output The number of "Line text" characters (out of 253) that replace the old string starting from the first character. If you specify "-1", trailing blanks will be removed and the number of characters used will be returned.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#LINE-NUMBER := 1          /*Replaces line 1
#COLUMN-FROM := 1
#COLUMN-TO := 0
#LINE-TEXT := 'New text'
PROCESS GUI ACTION EDIT-LINE-SET-TEXT WITH #EA-1 #LINE-NUMBER #COLUMN-FROM
#COLUMN-TO #LINE-TEXT #LINE-LENGTH GIVING #RESPONSE
MOVE #LINE-TEXT TO #EA-1.STRING
```

## GET-FOCUS Action

### Description

Gets the handle of the dialog element which currently has the focus, that is, which responds to keyboard input.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Output The dialog element that has the focus.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION GET-FOCUS WITH #FOCUS GIVING #RESPONSE
```

## GET-MESSAGE-TEXT Action

### Description

Gets the text for a given application message file number.

### Parameters

Name/Data Type	Explanation
Message number (I4)	Input The message number for which the text is to be read.
Destination (A253)	Output The message text.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION GET-MESSAGE-TEXT WITH #MESSAGENR #DEST GIVING #RESPONSE
```

## HELP Action

### Description

Invokes the help system and points to the given help topic in the help file HELP-FILENAME determined by the attribute for the specified dialog, or "*libraryname.hlp*" if this has not been set. Natural expects the help file to be located in one of the directories listed in the description of the HELP-FILENAME attribute.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The dialog or dialog element acting as parent window for the help window. Must not be an item. This parameter is also used to retrieve any non-default help file name via dialogs HELP-FILENAME attribute.
Help ID (I4)	Input The help topic ID.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#EA-1.HELP-ID := 1234
/*Set 1234 as help topic ID for the edit area
PROCESS GUI ACTION HELP WITH #EA-1 #EA-1.HELP-ID GIVING #RESPONSE
```

## HOURGLASS-REMOVE Action

### Description

Sets the pointer shape from "hourglass" to "arrow" and clears the pointer stack. You can also use this action to suspend the hourglass pointer and use the arrow shape while displaying a message box during long periods of processing. You resume with the hourglass pointer on the same nesting level as when it was suspended by passing the value returned in the level parameter to the HOURGLASS-STACK action.

### Parameters

Name/Data Type	Explanation
Level (I4)	Output The pointer stack level before the stack is cleared.

### Example:

```
PROCESS GUI ACTION HOURGLASS-REMOVE WITH #LEVEL GIVING #RESPONSE
```

# HOURLASS-STACK Action

## Description

Sets the pointer shape to "hourglass" and keeps the previous pointer shape in the stack of pointers. This indicates that end-user input is disabled during a long period of processing. To restore the previous shape (hourglass or arrow), you use the HOURLASS-UNSTACK action. This stack/unstack logic can be used in nested program structures to determine the pointer shape.

The level parameter can be used to resume the hourglass on the same stack level as when it was suspended by the HOURLASS-REMOVE action.

**Note:**

While a Natural dialog is performing a lengthy operation during which it is not able to process events, it should display the hourglass to the user to indicate that it is currently not available for operation. While the hourglass is active, no events will be delivered to the dialog.

## Parameters

Name/Data Type	Explanation
Level (I4)	Output The pointer stack level (max. 10) up to which the stack is filled with hourglass pointers.

**Note:**

The "Level" parameter is optional.

**Example:**

```
PROCESS GUI ACTION HOURLASS-STACK WITH #LEVEL GIVING #RESPONSE
```

## HOURGLASS-UNSTACK Action

### Description

Restores the previous pointer shape (hourglass or arrow) by decreasing the stack of pointers by one and by setting the pointer to the new stack top. If the stack is empty, the arrow shape is activated. This action corresponds to action HOURGLASS-STACK.

### Parameters

Name/Data Type	Explanation
Level (I4)	Output The current pointer stack level.

#### Note:

The "Level" parameter is optional.

#### Example:

```
PROCESS GUI ACTION HOURGLASS-UNSTACK WITH #LEVEL GIVING #RESPONSE
```

## INPUT-COPY-SELECTION Action

### Description

The text currently selected in the input dialog element is copied to the clipboard. The text can be selected in an input-field control, a selection-box control or an edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an input-field control, a selection-box control or an edit area control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-COPY-SELECTION WITH #IF-1 GIVING #RESPONSE
```

### Note:

For selection-boxes, the action must be called whilst the selection-box still has the focus.

## INPUT-CUT-SELECTION Action

### Description

The text currently selected in the input dialog element is copied to the clipboard and is deleted in the original place. The text can be selected in an input-field control, a selection-box control or an edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an input-field control, a selection-box control or an edit area control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-CUT-SELECTION WITH #IF-1 GIVING #RESPONSE
```

### Note:

For selection-boxes, the action must be called whilst the selection-box still has the focus.

## INPUT-DELETE-SELECTION Action

### Description

The text currently selected in the input dialog element is deleted. The text can be deleted from an input-field control, a selection-box control or an edit area control.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an input-field control, a selection-box control or an edit area control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-DELETE-SELECTION WITH #IF-1 GIVING #RESPONSE
```

### Note:

For selection-boxes, the action must be called whilst the selection-box still has the focus.

## INPUT-GET-LINE-LENGTH Action

### Description

Gets the length of a line in an edit area control or in an input-field control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an edit area control or an input-field control.
Line number (I4)	Input Length is retrieved from this line.
Line length (I4)	Output Returns the line length.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#LINE-NUMBER := 1 /* Examine the first line
PROCESS GUI ACTION INPUT-GET-LINE-LENGTH WITH #EA-1 #LINE-NUMBER #LINE-LENGTH
GIVING #RESPONSE
#IF-1.STRING := #LINE-LENGTH /* Display the result in this input-field control
```

## INPUT-GET-SELECTION Action

### Description

Retrieves the position of the selected text in an edit area control or in an input-field control from the beginning (character number) to the end (character number).

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an edit area control or an input-field control.
Position from (I4)	Output Selection starts from this position onwards.
Position to (I4)	Output Last selected position.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-GET-SELECTION WITH #EA-1 #POSITION-FROM #POSITION-TO
GIVING #RESPONSE
#IF-1.STRING := #POSITION-FROM /* Display the result in these two
#IF-2.STRING := #POSITION-TO /* input-field controls
```

## INPUT-GET-TEXT Action

### Description

Retrieves text in a line of the edit area control or in an input-field control (from a position onwards).

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an edit area control or an input-field control.
Position-from (I4)	Input The text is retrieved from this position onwards.
Line length (I4)	Input/Output The number of the characters to be retrieved (must be 1 to 253).
Line text (A253)	Output The retrieved text will be copied into this text string.
End (L)	Output Becomes TRUE if the last retrieved character was the last character in the dialog element. If "End" is TRUE, the number of retrieved characters is returned in the "Line length" parameter.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#POSITION := 8      /* Set input
#LINE-LENGTH := 22 /* values
PROCESS GUI ACTION INPUT-GET-TEXT WITH #EA-1 #POSITION-FROM #LINE-LENGTH
#LINE-TEXT #END GIVING #RESPONSE
#IF-1.STRING := #POSITION /* Display output values
#IF-2.STRING := #LINE-LENGTH /* in input-field controls
#IF-3.STRING := #TEXT
```

## INPUT-PASTE Action

### Description

Copies the content of the clipboard into an input-field control, a selection-box control, or an edit area control. If nothing is selected inside the dialog element, the clipboard text is inserted at the cursor position. If an area of text is selected, this selection is deleted and the content of the clipboard is inserted.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a selection-box control or an edit area control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-PASTE WITH #IF-1 GIVING #RESPONSE
```

## INPUT-SET-SELECTION Action

### Description

Selects an area (position to position) in an edit area control or in an input-field control.

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an edit area control or an input-field control.
Position from (I4)	Input Selection starts from this position onwards.
Position to (I4)	Input Last selected position. If you specify "0", the selection will extend to the last character.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#POSITION-FROM := 10 /* Select from the 10th to the 22nd character
#POSITION-TO := 22
PROCESS GUI ACTION INPUT-SET-SELECTION WITH #EA-1 #POSITION-FROM #POSITION-TO
GIVING #RESPONSE
```

## INPUT-SET-TEXT Action

### Description

Replaces text in an edit area control or in an input-field control (from position to position).

### Parameters

Name/Data Type	Explanation
HANDLE OF EDITAREA or INPUTFIELD	Input Specifies an edit area control or an input-field control.
Position-from (I4)	Input Position of the first character to be replaced. If you specify "0" the text is appended to the text already present in the dialog element.
Position-to (I4)	Input/Output Position of the last character to be replaced. If you specify "0", the index of the last character in the dialog element will be returned.
Text length (I4)	Input/Output The number of characters to be inserted before "Position-to" starting from the first character of "Line text". If you specify "-1", the "Line text" string will be inserted into the new line, trailing blanks will be removed and the number of copied characters will be returned.
Text (A253)	Input This text string replaces the old one.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#POSITION-FROM := 8
#POSITION-TO := 24
#TEXT-LENGTH := 16
#TEXT := 'Insert this text'
PROCESS GUI ACTION INPUT-SET-TEXT WITH #EA-1 #POSITION-FROM #POSITION-TO
#TEXT-LENGTH #TEXT GIVING #RESPONSE
```

## INPUT-UNDO Action

### Description

Undoes the last editing action in an input-field control, a selection-box control or an edit area control. Editing actions that can be undone are entering, copying, pasting, cutting and deleting text in a dialog element.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an input-field control, a selection-box control or an edit area control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INPUT-UNDO WITH #EA-1 GIVING #RESPONSE
```

## INQ-CLICKPOSITION Action

### Description

Returns the x-axis and the y-axis position where the end user has clicked (relative to the dialog element's rectangle). These coordinates are also updated immediately before a context-menu's before-open event is called. The application can then combine this call with a call to either INQ-ITEM-BY-POSITION or TABLE-INQUIRE-CELL to determine which list box item or table control cell was "right-clicked".

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a dialog element.
X-Position (I4)	Output The x-axis position in pixels.
Y-Position (I4)	Output The y-axis position in pixels.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INQ-CLICKPOSITION WITH #BM-1 #X-POSITION #Y-POSITION
GIVING #RESPONSE
#IF-1.STRING := #X-POSITION /* Display the coordinates in these two
#IF-2.STRING := #Y-POSITION /* input-field controls
```

## INQ-DRAG-DROP Action

### Description

Can be used in a drag-drop event of a bitmap control to retrieve the position where the dropping occurred. The parameters X-Position and Y-Position return the coordinates of the position in pixels relative to the upper left corner of the target bitmap control. The Mode parameter indicates whether SHIFT and/or CTRL was pressed while the dropping occurred.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Output The handle of the dialog in which the bitmap control was dropped.
HANDLE OF GUI	Output The dragged bitmap control's handle.
Mode (I4)	Output Possible values: 0 no key. 1 SHIFT key. 2 CTRL key. 3 SHIFT+CTRL keys.
X-Position (I4)	Output X-axis position on the target.
Y-Position (I4)	Output Y-axis position on the target.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INQ-DRAG-DROP WITH #DIA-HANDLE #DIA-ELEMENT #MODE
#X-POSITION #Y-POSITION GIVING #RESPONSE
#IF-1.STRING := #X-POSITION /* Display the coordinates in these two
#IF-2.STRING := #Y-POSITION /* input-field controls
```

## INQ-INNER-RECT Action

### Description

Provides the width and the height of the client rectangle (in pixels) in a dialog window or dialog element. For dialogs, this is the area into which dialog elements can be placed. Dialog elements cannot be placed on the decoration, the title bar, a tool bar, a menu bar, a status bar, and the horizontal and vertical scroll bars. For dialog elements, this is the area excluding any frame components such as borders and scroll bars.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element that contains the client rectangle. If the handle value is NULL-HANDLE, the width and height of the screen is returned.
Width (I4)	Output The width of the client rectangle.
Height (I4)	Output The height of the client rectangle.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION INQ-INNER-RECT WITH #DLG$WINDOW #WIDTH #HEIGHT GIVING
#RESPONSE
#IF-1.STRING := #HEIGHT /* Display the width and height
#IF-2.STRING := #WIDTH
```

## INQ-ITEM-BY-POSITION Action

### Description

Returns the item at a given position within a control. This is particularly useful for finding out which list box item or status bar pane was clicked with the right mouse button before a context menu is displayed (see example below).

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies a list box or status bar control.
X-Position (I4)	Input Specifies the x-axis position in pixels relative to the top-left hand corner of the control.
Y-Position (I4)	Input Specifies the y-axis position in pixels relative to the top-left hand corner of the control.
Item (HANDLE OF GUI)	Output The handle of the list box item or status bar pane at the specified position, or NULL-HANDLE if none.
Response (I4)	Output NATURAL error (if applicable).

### Example:

```

/* Sample BEFORE-OPEN code for context menu:
PROCESS GUI ACTION INQ-CLICKPOSITION WITH
#LB-1 #X-POSITION #Y-POSITION GIVING #RESPONSE
PROCESS GUI ACTION INQ-ITEM-BY-POSITION WITH
#LB-1 #X-POSITION #Y-POSITION #LBITEM GIVING #RESPONSE

```

## LOAD-LAYOUT Action

### Description

Loads the specified dialog's control bar layout (if any) that was previously saved for the current user via the SAVE-LAYOUT action. If no such saved layout exists, this action has no effect.

Note that the bar types (dockable or fixed) should match those at the time the layout was saved, otherwise the results are unpredictable.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog for which the control bar layout is to be loaded.
Profile name (A253)	Input (optional parameter) The name of the profile from which the information is to be loaded. If this parameter is not specified, the name of the library containing the dialog is used.
Section name (A253)	Input (optional parameter) The name of the profile section from which the information is to be loaded. If this parameter is not specified, the name of the dialog is used.

### Example:

```
PROCESS GUI ACTION LOAD-LAYOUT WITH #DLG$WINDOW GIVING #RESPONSE
```

## MOVE-NAVIGATION-ITEMS Action

### Description

Moves a range of controls to a new position in the control sequence. The control sequence within the range of controls being moved remains unchanged.

### Parameters

Name/Data Type	Explanation
First control (HANDLE OF GUI)	Input Specifies the first control in the range of controls which is to be moved. If this parameter is specified as NULL-HANDLE, the first control in the control sequence is used.
Last control (HANDLE OF GUI)	Input Specifies the last control in the range of controls which is to be moved. If this parameter is specified as NULL-HANDLE, the last control in the control sequence is used.
Position To (HANDLE OF GUI)	Input Specifies the control which is to immediately precede the moved control(s) in the new control sequence. This control must not be one of the controls being moved. If this parameter is specified as NULL-HANDLE, the controls are moved to the end of the control sequence. If this parameter is specified as the handle of the dialog itself, the controls are moved to the front of the control sequence.
Response (I4)	Output NATURAL error (if applicable).

### Example:

```

/* Move the controls in the current control sequence starting from #PB-1 and ending with the last
/* control in the control sequence to follow #SB-1
PROCESS GUI ACTION MOVE-NAVIGATION-ITEMS WITH
    #PB-1 NULL-HANDLE #SB-1 GIVING #RESPONSE

```

# OLE-ACTIVATE

## Description

Activates the default action of an OLE server programmatically. To enable the end user to activate the OLE server, you set the OLE container control's MODIFIABLE attribute to TRUE. The end user then activates the OLE server by using the right mouse button to click in the OLE container control's rectangle; this shows the context menu, from which the end user can choose the appropriate command. The OLE server can also be activated by using the left mouse button to double-click in the OLE container control's rectangle. If the menu item "Show Object Verbs" is provided, the end user can also use this to activate the server.

## Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
Response (I4)	Output Natural error (if applicable).

## Example:

```
PROCESS GUI ACTION OLE-ACTIVATE WITH #OCT-1 GIVING #RESPONSE
```

# OLE-DEACTIVATE

## Description

Deactivates an OLE server programmatically, that is, in-place editing is finished. The end user can also deactivate the server by pressing ESC or by clicking into the Natural dialog outside the OLE container control.

## Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
Response (I4)	Output Natural error (if applicable).

## Example:

```
PROCESS GUI ACTION OLE-DEACTIVATE WITH #OCT-1 GIVING #RESPONSE
```

## OLE-GET-DATA

### Description

Reads an embedded object into a Natural variable. It is recommended to define an array of variables that is large enough to hold the object: some objects are at least 10 KB in size. You then get the object by executing this procedure; the second parameter contains the size of the array. The procedure returns the real size being used and a Natural error code other than "0" is returned. To avoid getting this error code, you first query the current size of the object with the OBJECT-SIZE attribute.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
Variable	Input Variable or (usually) an array of variables.
Size (I4)	Input Size of the variable or the array of variables provided.
Real size (I4)	Output Real size of the variable or the array of variables.
Response (I4)	Output Natural error (if applicable).

### Example:

```
#CURRSIZE := #OCT-1.OBJECT-SIZE /* How large is the object?
IF.. /* If the object is too large, do the following...
... /* If not, execute the procedure
END-IF
PROCESS GUI ACTION OLE-GET-DATA WITH #OCT-1 #MYVARI (1:5) 15000 #REALSIZE GIVING
#RESPONSE
```

## OLE-INSERT-OBJECT

### Description

Provides the end user with a dialog box which allows him/her to select and start an OLE server application or an external object.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
String (A253)	Input Dialog box caption.
Flag (I4)	Input For future use.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION OLE-INSERT-OBJECT WITH #MYCONTAINER 'My Caption' #FLAG GIVING  
#RESPONSE
```

## OLE-READ-FROM-FILE

### Description

Reads an embedded object into the OLE container control.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
File name (A253)	Input Specifies the name of an embedded object (in Natural, this is a ".neo" file)
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION OLE-READ-FROM-FILE WITH #MYCONTAINER 'c:\natgui\myobject.neo'  
GIVING #RESPONSE
```

## OLE-SAVE-TO-FILE

### Description

Saves the currently embedded object to a file with the default extension ".neo" (meaning "Natural Embedded Object").

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
File name (A253)	Input Specifies the name of an embedded object (in Natural, this is a ".neo" file)
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION OLE-SAVE-TO-FILE WITH #MYCONTAINER 'MYOBJECT.NEO' GIVING  
#RESPONSE
```

# OLE-SET-DATA

## Description

Puts the content of a Natural variable into an embedded object. Corresponds with the OLE-GET-DATA action. You put the variable content into the object by executing this procedure; the second parameter contains the size of the object. It is recommended to define an array of variables that is large enough to hold the object: some objects are at least 10 KB in size.

## Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Specifies an OLE container control.
Variable	Input Variable or (usually) an array of variables.
Size (I4)	Input Size of the object.
Response (I4)	Output Natural error (if applicable).

## Example:

```
PROCESS GUI ACTION OLE-SET-DATA WITH #OCT-1 #MYVARI (1:5) #REALSIZE GIVING
#RESPONSE
```

## PICK-FILENAME Action

### Description

Invokes a standard dialog box that allows the end user to pick the name of a file which is **optionally** assigned to a Natural work file number.

### Parameters

Name/Data Type	Explanation
Title (A253)	Input The text displayed in the title bar of the dialog box.
Init (A253)	Input A string pattern ( <i>[directoryname][filename[.extension]]</i> ) that determines the initial directory (if specified), together with either the files to choose from (if the <i>filter list</i> parameter is not specified) or the pre-selected file (if a filename and extension containing no wildcard characters (*,?) is used). If the pattern is an empty string, all files of the current directory are listed.
Work file number (I4)	Input The work file number to which the selected file name is to be assigned. If this parameter is "0", the file name is not assigned to a work file number.
File name (A253)	Output Returns the selected file name with the entire path name.
Dialog Type (I4)	Input (optional parameter) Determines type of file selection dialog displayed: 0 (default) = 'Open' dialog (new files allowed); 1 = 'Open' dialog (selected file must exist); 2 = 'Save' dialog.
Filter list (A253/*)	Input (optional parameter) A one-dimensional array containing pre-specified display filter definitions. The array does not need to be filled entirely. Each filter definition is represented by a <i>pair</i> of array elements. The first element of each pair is a textual description of the filter used for display in the file selection dialog. The second element of each pair is the associated display filter pattern. The display filter pattern may be complex, consisting of multiple components separated by a semi-colon.
Filter index (I4)	Input (optional parameter) The index of the filter to be initially used when the file selection dialog is first opened. 0 = first filter, 1 = second filter, and so on.
Response (I4)	Output Natural error (if applicable).

**Example:**

```
DEFINE DATA LOCAL
1 #RESPONSE (I4)
1 #MYTITLE (A253)
1 #SELECTNAME (A253)
1 #FILTER (A32/6) CONST
  <
    'Private resource files', '*.nr*',
    'Shared resource files', '*.bmp;*.ico;*.hlp;*.neo;*.rpt',
    'All Files (*.*)', '*.*'
  >
END-DEFINE
...
#MYTITLE := 'Example using implicit display filter'
PROCESS GUI ACTION PICK-FILENAME WITH #MYTITLE 'c:\*.*' 0
#SELECTNAME GIVING #RESPONSE
#IF-1.STRING := #SELECTNAME
*
#MYTITLE := 'Example using explicit display filter'
PROCESS GUI ACTION PICK-FILENAME WITH #MYTITLE 'd:\fuser\mylib\res\'
  0 #SELECTNAME 0 #FILTER(*) 1 GIVING #RESPONSE
#IF-2.STRING := #SELECTNAME
```

## PLAY-SOUND Action

### Description

Plays a tune from a ".WAV" audio file.

**Note:**

This action is not available under Windows NT.

### Parameters

Name/Data Type	Explanation
Audio file name (.wav)	Input Either the full path name of the audio file or just the file name. If you specify just the file name, this file must be located in the directory referred to by the environment variable NATGUI_BMP. If you specify a path name, this must be the fully expanded path name including the drive and the path specifications.

**Examples:**

```
PROCESS GUI ACTION PLAY-SOUND WITH 'CHIMES.WAV'  
PROCESS GUI ACTION PLAY-SOUND WITH #FILE GIVING #RESPONSE
```

## PROCESS-EVENTS Action

### Description

Processes all outstanding messages for the Natural application (raising any associated events) before returning. This is particularly useful for allowing the user to interrupt a long operation. Typically, you create a simple abort dialog of style 'modal' with a 'Cancel' push button, and call this action periodically whilst processing the long operation. If the user has clicked the 'Cancel' push button in the meantime, the CLICK event for this button will be received during the call, which typically sets an abort flag that can be tested on return and handled accordingly.

#### Note:

This action must be used with care, because ALL user interactions with the application will be processed during the call. This includes clicks on tool bar or menu items, for instance, unless these are explicitly disabled, or if a modal dialog is currently on display (as in the above example).

### Parameters

Response (I4)	Output Natural error (if applicable).
---------------	--

#### Example:

```
PROCESS GUI ACTION PROCESS-EVENTS GIVING #RESPONSE
```

## RECALC-LAYOUT Action

### Description

Causes the control bar layout for the specified dialog to be recalculated, which may result in the bar positions and/or sizes being adjusted. The control bars include the newer tool-bar and status-bar controls, but not the traditional tool bar and status bar available in earlier Natural versions.

Normally, Natural implicitly causes a layout recalculation where necessary (e.g. immediately before the after-open event, or if a bar is created or moved). However, in order to reduce the number of screen refreshes to a minimum, it may be that only a delayed implicit layout recalculation is performed. Since some operations, such as getting the dialog's client area via the INQ-INNER-RECT action, require that the bars are in their final positions, a preceding manually-triggered layout recalculation may be necessary in some cases. Note that this is not the case for all bars defined statically with the dialog editor, because Natural automatically does an immediate layout recalculation before calling the user's AFTER-OPEN event handler.

Note that an immediate layout recalculation will clear any outstanding delayed layout recalculation request. Also, multiple delayed layout recalculation requests are consolidated (i.e., do not cause multiple refreshes).

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog whose bar layout is to be recalculated.
Mode (L)	Input (optional parameter) If true (default), an immediate layout recalculation is performed. If false, a delayed layout recalculation is performed.

### Example:

```
PROCESS GUI ACTION RECALC-LAYOUT WITH #DLG$WINDOW GIVING #RESPONSE
```

## REFRESH-LINKS Action

### Description

Refreshes the content of those input-field controls and selection-box controls for which the linked variable option was chosen for the content variables. This becomes necessary after these variables have been changed in code to display new values, for example, after a new record has been read from a database.

#### Note:

If the content of such a dialog element is modified by the end user, the corresponding linked variable is updated automatically when the dialog element loses the focus.

### Parameters

Name/Data Type	Explanation
HANDLE OF dialog	Input The dialog in which all linked variables are to be refreshed.
Response (I4)	Output Natural error (if applicable).

#### Example:

```
#MYVARI := 'Refreshed'
PROCESS GUI ACTION REFRESH-LINKS WITH #DLG$WINDOW GIVING #RESPONSE
```

## RESET-ATTRIBUTES Action

### Description

Resets all attributes in the global attribute list to their default values. For the default values, see the section Attributes. It is recommended to use this action before creating dialog elements dynamically with the ADD action, for example, in an after open event handler.

### Parameters

Name/Data Type	Explanation
Response (I4)	Output Natural error (if applicable).

### Example:

```
/* After open event handler code
PROCESS GUI ACTION RESET-ATTRIBUTES GIVING #RESPONSE
PROCESS GUI ACTION ADD...
```

## SAVE-LAYOUT Action

### Description

Saves the specified dialog's current control bar layout for the current user. The control bars include the newer tool-bar and status-bar controls, but not the traditional tool bar and status bar available in earlier Natural versions. The layout includes the bar type (dockable or fixed), location, position, and the bar's visible and enabled state.

The information is currently stored in the registry under:

```
\\HKEY_CURRENT_USER\Software\Software AG\Natural Applications\<<profile name>\<section name>
```

where <profile name> and <section name> are described in the table below. However, the base location (or whether the information is stored in the registry at all) may change for future Natural versions.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog for which the control bar layout is to be saved.
Profile name (A253)	Input (optional parameter) The name of the profile to which the information is to be saved. If this parameter is not specified, the name of the library containing the dialog is used.
Section name (A253)	Input (optional parameter) The name of the profile section to which the information is to be saved. If this parameter is not specified, the name of the dialog is used.

### Example:

```
PROCESS GUI ACTION LOAD-LAYOUT WITH #DLG$WINDOW GIVING #RESPONSE
```

## SET-FOCUS Action

### Description

Selects a dialog element so that the end user can enter data with the keyboard. You can use this action with handles of the following dialog elements: bitmap control, edit area control, input-field control, list box control, push button control, radio button control, scroll bar control, selection-box control, toggle button control. You can also use this action with a dialog's window handle, for example, with "#DLG\$WINDOW".

### Note:

It is not recommended to execute the SET-FOCUS action from the leave event, because the leave event implies that the focus is to be set to yet another part of the application.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog element of the above-mentioned type that is supposed to get the focus.
Response (I4)	Output Natural error (if applicable).

**Example:**

```
#INPUT := #IF-1  
PROCESS GUI ACTION SET-FOCUS WITH #INPUT GIVING #RESPONSE
```

## SET-TABS Action

### Description

Sets tabulator stops in a list box control even if the font in the list box control is proportional. The tabulator stops apply to the STRING attribute in any item of the list box control.

### Parameters

Name/Data Type	Explanation
HANDLE OF LISTBOX	Input Handle of the list box control.
Tabstop (I4)	Input The tabstop is measured in units of one quarter of the system font's average width. You can enter this parameter any number of times (in ascending order). In the list box items' STRING, you determine the place of the tabstop by inserting the HORIZONTAL-TAB character (defined in the local data area NGULKEY1). Inserting this character will move the following STRING text to the next tabstop.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION SET-TABS WITH #LB-1 50 100 150 GIVING #RESPONSE
COMPRESS 'HELLO' HORIZONTAL-TAB 'HUGO' TO #STRING
PROCESS GUI ACTION ADD-ITEMS WITH #LB-1 1 #STRING
```

## SYSTEM-GET-NATIVE-HANDLE Action

### Description

Queries the native handle of a dialog element in terms of the windowing system. You can use this native handle for external function calls.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The handle of the dialog element in Natural terms.
Native handle (I4)	Output The window handle of the dialog element in terms of the windowing system. For font controls, the native font handle is returned. For menu bars, menu items and submenu controls, the native menu handle is returned. For bitmap controls, the native bitmap handle is returned. For all other dialogs and dialog elements, the native window handle is returned.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION SYSTEM-GET-NATIVE-HANDLE WITH #PB-1 #NATIVEHANDLE GIVING
#RESPONSE
#IF-1.STRING := #NATIVEHANDLE /* Display the handle value
```

## SYSTEM-PRINTER-SETUP Action

### Description

Invokes a dialog box that allows the end user to assign a physical printer to a logical device name. You can assign this logical device name to a report using the DEFINE PRINTER statement.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input The parent dialog element of the printer assignment dialog box.
Logical device (A8)	Input/Output The logical device name to which the printer is assigned. Possible values: "LPT1" to "LPT31", "lpt1" to "lpt31".
Commit (L)	Output (optional parameter) Returns TRUE if the user closed the dialog box via the 'OK' push button, or FALSE if the dialog was cancelled or if an error occurred.
Response (I4)	Output Natural error (if applicable).

#### Note:

The first two parameters are mandatory.

#### Example:

```
PROCESS GUI ACTION SYSTEM-PRINTER-SETUP WITH #PB-1 'LPT1' GIVING #RESPONSE
```

## TABLE-DELETE-ROW Action

### Description

Deletes a row in a table. To do so, one of the table control's STYLE attribute values must be "e". Specifying the value "0" in the Row index parameter is useful in combination with the delete-row event because the ROW attribute then contains the index of the row in which the end user has caused the most recent event.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Row Index (I4)	Input Specifies the number of the row to be deleted. If you specify "0", the value of the table's ROW attribute is taken as the number of the row to be deleted. If you specify "-1", the last row in the table will be deleted. If you specify "1", the first row in the table will be deleted.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-DELETE-ROW WITH #TBL-1 2 GIVING #RESPONSE
...
PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1 GIVING #RESPONSE /* Recommended
```

## TABLE-FIND-FIELD Action

### Description

Finds the column with the specified title.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Field name (A253)	Input The column title to be found.
Index (I4)	Output The index of the column where the title was found.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-FIND-FIELD WITH #TBL-1 FLD1 #IDX GIVING #RESPONSE
```

## TABLE-GET-SELECTION Action

### Description

Retrieves the column and row indices of the selected cells in a table control from the beginning of the first selected cell to the end of the last selected cell.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Column from (I4)	Output First selected ...
Row from (I4)	Output ...cell.
Column to (I4)	Output Last selected ...
Row to (I4)	Output ...cell.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-GET-SELECTION WITH #TBL-1 #COLFIRST #ROWFIRST #COLLAST
#ROWLAST GIVING #RESPONSE
```

## TABLE-INQUIRE-CELL Action

### Description

enables you to inquire which cell is at a particular pixel offset within the table. This is particularly useful for determining the cell for which a context menu was invoked, in which case the position specified should be that returned by the INQ-CLICKPOSITION action used within the context menu's BEFORE-OPEN event.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
X-Position (I4)	Input Specifies the x-axis position in pixels relative to the top-left hand corner of the table.
Y-Position (I4)	Input Specifies the y-axis position in pixels relative to the top-left hand corner of the table.
Row Index (I4)	Output The row corresponding to the specified y-axis position, or 0 if outside the range of any row.
Column Index (I4)	Output The column corresponding to the specified x-axis position, or 0 if outside the range of any column.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-INQUIRE-CELL WITH #TBL-1 #X-POSITION #Y-POSITION #ROW #COLUMN
GIVING #RESPONSE
```

## TABLE-INQUIRE-ROW Action

### Description

enables you to inquire the value in each cell of a single row. The values will update the input parameters you specify. Later at runtime, you can retrieve the updated parameter values. Specifying the value "0" in the "Row Index" parameter is useful in combination with the insert-row event because the ROW attribute then contains the index of the row in which the end user has caused the most recent event.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Row Index (I4)	Input Specifies the number of the row to be inquired. If you specify "0", the value of the table's ROW attribute is taken as the number of the row to be inquired. If you specify "-1", the last row in the table will be inquired. If you specify "1", the first row in the table will be inquired.
List of parameters	Input You can either specify one parameter for each column of the row, or you can specify less parameters than there are columns. If you specify more parameters than there are columns, an error message is returned.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-INQUIRE-ROW WITH #TBL-1 3 #P1 #P2 #P3 #P4
GIVING #RESPONSE
```

## TABLE-INSERT-ROW Action

### Description

Inserts a row into a table. To do so, one of the table control's STYLE attribute values must be "e". Specifying the value "0" in the "Row Index" parameter is useful in combination with the insert-row event because the ROW attribute then contains the index of the row in which the end user has caused the most recent event.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Row Index (I4)	Input Specifies the number of the row before which the new row is to be inserted. If you specify "0", the value of the table's ROW attribute is taken as the number of the row before which the new row is to be inserted. If you specify "-1", the new row will be appended to the last row in the table. If you specify "1", the row to be inserted will become the first row in the table.
List of parameters	Input You can either specify one parameter for each column of the row, or you can specify less parameters than there are columns. If you specify more parameters than there are columns, an error message is returned.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-INSERT-ROW WITH #TBL-1 3 #P1 #P2 #P3 #P4
GIVING #RESPONSE
PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1 GIVING #RESPONSE /* Recommended
```

## TABLE-REFRESH Action

### Description

Refreshes the contents of a table control.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-REFRESH WITH #TBL-1 GIVING #RESPONSE
```

## TABLE-SET-SELECTION Action

### Description

Selects a rectangular range of cells in a table control.

### Parameters

Name/Data Type	Explanation
HANDLE OF TABLE	Input Specifies a table control.
Column from (I4)	Input First selected ...
Row from (I4)	Input ...cell.
Column to (I4)	Input Last selected ...
Row to (I4)	Input ...cell. If you specify "0", the selection will extend to the last cell in the table.
Response (I4)	Output Natural error (if applicable).

### Example:

```
PROCESS GUI ACTION TABLE-SET-SELECTION WITH #TBL-1 #COLFIRST #ROWFIRST #COLLAST
#ROWLAST GIVING #RESPONSE
```

## TEXT-GET-EXTENT Action

### Description

Returns the size of a text if a certain font is chosen. You can use this action, for example, to find out the best font that still has an acceptable width and height.

### Parameters

Name/Data Type	Explanation
HANDLE OF FONT	Input Specifies a font control.
Text (A253)	Input Your text.
Width (I4)	Output Width in pixels.
Height (I4)	Output Height in pixels.
Response (I4)	Output Natural error (if applicable).

### Example:

```

/*Check the size a font will require
#TEXT := 'Check the size'
PROCESS GUI ACTION TEXT-GET-EXTENT WITH FONT-HANDLE #TEXT #WIDTH #HEIGHT
GIVING #RESPONSE

```

## UPDATE-COMMAND-STATUS Action

### Description

A dialog automatically receives command-status events (unless suppressed) during idle processing when the status of one or more commands (signals, or menu items or tool bar items without a SAME-AS attribute) need to be updated. This action, however, allows the same updating logic to be triggered manually at any time, in addition to the automatic idle-time update.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog or dialog element whose status is to be updated, or NULL-HANDLE. If a dialog handle is specified, all commands belonging to the specified dialog (excluding submenus and context menus) are included in the updating process. If a dialog element handle is specified, all commands belonging to the specified dialog element are included in the updating process. If NULL-HANDLE is specified, all commands in all dialogs (excluding submenus and context menus) are included in the updating process, as is the case for the implicit idle-time update.

### Example:

```
PROCESS GUI ACTION UPDATE-COMMAND-STATUS WITH NULL-HANDLE GIVING #RESPONSE
```

## VALIDATE Action

### Description

Validates a handle variable. A handle variable is considered valid if it contains either the value NULL-HANDLE or a handle to an existing dialog or dialog element.

### Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Any dialog or dialog element.
Response (I4)	Output Natural error (if applicable), or 0 if handle variable is valid.

### Example:

```
PROCESS GUI ACTION VALIDATE WITH #PB-1 GIVING #RESPONSE
```

# NGU Subprograms and Dialogs

---

## General Information

The NGU-prefixed subprograms and dialogs in library SYSTEM provide you with frequently needed functionality. You then only specify a CALLNAT or OPEN DIALOG statement in your event handler code, instead of having to program everything manually.

For your convenience, the local data areas NGULKEY1 and NGULFCT1 are automatically included in the list of local data areas used by any new dialog.

- NGULFCT1 lists the names of the subprograms and dialogs as such.
- NGULKEY1 lists reserved symbols to be used in any event handler code. This enables you to refer to certain attribute values by the more meaningful texts rather than by the integer values. It also enables you to use meaningful dialog element names as parameters in a CALLNAT to an NGU-prefixed subprogram or in an OPEN DIALOG to an NGU-prefixed dialog.

There are two types of names for the NGULFCT1 subprograms or dialogs:

- Long Name (explanatory) is the basis of the alphabetical list.
- Natural Object Name is the shorter name.

To call the subprograms, you can use either name in event handler code: you specify CALLNAT *subprogram-long-name/Natural-object-name parameter-name... .*

To call the dialogs, you can also use both types of names inside the OPEN DIALOG statement.

**Note:** The NGU subprograms and dialogs provided with Natural Version 2.1.3 have partly been replaced by corresponding PROCESS GUI statement actions. The old NGU subprograms and dialogs are compatible with this version. It is recommended that you replace the calls to the dialogs and subprograms with the corresponding PROCESS GUI statement actions. This will increase the performance of your applications.

## NGULKEY1 Reserved Symbols

### Color Symbols

DEFAULT (0)  
WHITE (1)  
BLACK (2)  
LTGREY (3)  
GREY (4)  
DKGREY (5)  
RED (6)  
GREEN (7)  
BLUE (8)  
CYAN (9)  
MAGENTA (10)  
BROWN (11)  
YELLOW (12)  
LIGHTRED (13)  
LIGHTGREEN (14)  
LIGHTBLUE (15)  
LIGHTCYAN (16)

LIGHTMAGENTA (17)  
 BRIGHTWHITE (18)  
 CUSTOM (50)

## Dialog Element Types

BITMAP  
 CANVAS  
 COLUMNSPECIFICATION  
 CONTEXTMENU  
 CONTROLBOX  
 EDITAREA  
 FONT  
 GRAPHICTEXT  
 GROUPFRAME  
 INPUTFIELD  
 LINE  
 LISTBOX  
 LISTBOXITEM  
 MDICHILD  
 MDIFRAME  
 MENUBAR  
 MENUITEM  
 PUSHBUTTON  
 RADIOBUTTON  
 RECTANGLE  
 SCROLLBAR  
 SELECTIONBOX  
 SELECTIONBOXITEM  
 SUBMENU  
 TABLE  
 TIMER  
 TEXTCONSTANT  
 TOGGLEBUTTON  
 TOOLBAR  
 TOOLBARITEM  
 WINDOW

## Event-Suppressing Symbols

NOT-SUPPRESSED  
 SUPPRESSED

## Menu Item Style Symbols

MT-NORMAL  
 MT-SEPARATOR  
 MT-SUBMENU  
 MT-MDICASCADE  
 MT-MDITILE  
 MT-MDIARRANGE  
 MT-WINDOWMENU  
 MT-CUT  
 MT-COPY  
 MT-PASTE  
 MT-DELETE

MT-UNDO

## Menu Item Symbols

UNCHECKED

CHECKED

## Separator Symbols

END-OF-LINE

For details on END-OF-LINE, see the section NGU-MESSAGEBOX Dialog.

HORIZONTAL-TAB

For details on HORIZONTAL-TAB, see the section SET-TABS Action.

## Tool Bar Symbols

TB-TOP

TB-BOTTOM

TB-LEFT

TB-RIGHT

## NGU-CLIENT-ADVISE-HOT Subprogram

### Natural Object Name

NGUCLADH

### Description

Requests the server identified by DDE-VIEW.CONV-ID to advise the client as soon as new data are available. If so, the client will be advised of the new data. This will be specified in DDE-VIEW.ITEM. The data will then be sent automatically.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-ADVISE-TERM Subprogram

### Natural Object Name

NGUCLADT

### Description

Requests the server to no longer advise the client of new data.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-ADVISE-WARM Subprogram

### Natural Object Name

NGUCLADW

### Description

Requests the server identified by DDE-VIEW.CONV-ID to advise the client as soon as new data are available. If so, the client will only be advised of the new data. This will be specified in DDE-VIEW.ITEM. If the application wants the data, it will have to request the data.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-CONNECT Subprogram

### Natural Object Name

NGUCLCON

### Description

This subprogram requests a conversation with a server providing the service DDE-VIEW.SERVICE on the topic DDE-VIEW.TOPIC. If an appropriate server can be found, this starts a conversation identified by a unique conversation ID, which is returned to DDE-VIEW.CONV-ID. If not, this field is empty on return of the request. If you plan on having more than one conversation, store the value of DDE-VIEW.CONV-ID as you will need it to identify conversations.

Before you execute this subprogram, you must assign the \*DIALOG-ID system variable of the client dialog to DDE-VIEW.CONV-ID. You must also assign the string "DLGID" to DDE-VIEW.MESSAGE.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-DISCONNECT Subprogram

### Natural Object Name

NGUCLDIS

### Description

Closes the conversation identified by DDE-VIEW.CONV-ID. The server will be advised of this.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-EXECUTE Subprogram

### Natural Object Name

NGUCLEXE

### Description

Requests the server identified by DDE-VIEW.CONV-ID to execute a command. The field DDE-VIEW.DATA describes the command to be executed.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-GET-DATA Subprogram

### Natural Object Name

NGUCLGDA

### Description

You must use this subprogram in the event handler of a DDE-client event to fill the DDE-VIEW parameter data area. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE. The possible values of DDE-VIEW.MESSAGE are:

<b>DISCONNECT</b>	The server has terminated the conversation identified by DDE-VIEW.CONV-ID by unregistering the topic, or by exiting.
<b>DATA</b>	The server has sent data for DDE-VIEW.CONV-ID and DDE-VIEW.ITEM. The data are contained in DDE-VIEW.DATA. The length of the data will be contained in DDE-VIEW.DATALEN.
<b>NOTIFY</b>	The server has new data available for the conversation identified by DDE-VIEW.CONV-ID and DDE-VIEW.ITEM. The data are not sent at this point. You must request these data in your event handler. Only then the data will actually be received.
<b>TIMEOUT</b>	No server message has been received during the timeout period.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

# NGU-CLIENT-POKE Subprogram

## Natural Object Name

NGUCLPOK

## Description

Sends data to the server identified by DDE-VIEW.CONV-ID. These data will be specified in DDE-VIEW.ITEM. When sending data to the server, you must set DDE-VIEW.DATALEN to a value:

- If set to "0", the entire data will be sent; the server will send on return a value that indicates how many data were actually sent.
- If set to "-1", trailing blanks will be removed and the server will send on return a value that indicates how many data were actually sent.
- If set to a value greater zero, this amount of data is actually sent.

The data to be sent are contained in DDE-VIEW.DATA.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-REQUEST Subprogram

### Natural Object Name

NGUCLREQ

### Description

Requests data from the server identified by DDE-VIEW.CONV-ID. These data will be specified in DDE-VIEW.ITEM. If data are available, the server will send them to the client.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-CLIENT-STOP Subprogram

### Natural Object Name

NGUCLSTP

### Description

Terminates all DDE activity for the client. Any open conversations are terminated. This subprogram may be useful to handle an error returned from a client request.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-COLOUR-SELECT Dialog

### Natural Object Name

NGUCOSE1

### Description

Provides a dialog box for selecting or creating a color.

### OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF parent	USING clause/Input The parent dialog.
Colour name (I4)	WITH clause/Output Returns a color.
Colour value (B3)	WITH clause/Output Returns an RGB-defined color.
Colour selected (L)	WITH clause/Output Indicates whether a color has been selected.
Style (A32)	WITH clause/Output Reserved for future use.

### Example:

```
OPEN DIALOG NGU-COLOUR-SELECT
  USING #DLG$WINDOW
  GIVING #DLG /* The parent's handle value (I4)
  WITH #COLOUR-NAME #COLOUR-VALUE #COLOUR-SEL #STYLE
```

# NGU-DIALOG-CLOSE-ALL Subprogram and Subroutine

## Natural Object Names

NGUDICL1  
 NGUDICLS

## Description

Both the subprogram and the subroutine close all child dialogs of a given dialog. If you execute them, the close event occurs for all child dialogs. You can use them, for example, to close all MDI child dialogs in an MDI application. The difference between the two is that you use the NGUDICLS subroutine if you are accessing a global data area while closing all child dialogs. NGUDICLS is to be found in library SYSNGU. If the dialog calling this subprogram/subroutine is an MDI child dialog itself and all MDI child dialogs are to be closed, you must specify the parent of this dialog (the MDI frame dialog) as parameter. The MDI child dialog calling this subprogram/subroutine will then be the last to be closed.

## Parameters

Name/Data Type	Explanation
HANDLE OF GUI	Input Handle of the dialog whose child dialogs are to be closed.
Response (I4)	Output Natural error (if applicable).

## Examples:

```
CALLNAT NGU-DIALOG-CLOSE-ALL #MYDIA #RESPONSE /* NGUDICL1 subprogram
PERFORM NGU-DIALOG-CLOSE-ALL #MYDIA #RESPONSE /* NGUDICLS subroutine
```

## NGU-FONT-SELECT Dialog

### Natural Object Name

NGUFOSE1

### Description

Provides the end user with a dialog box to select a font. To avoid problems, always ensure that you assign NULL-HANDLE to your font before you specify the OPEN DIALOG statement. Otherwise, an invalid font handle is assigned to the text(s) for which the end user has already selected a font.

### OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF GUI	USING clause/Input The parent dialog of the font selection dialog box.
HANDLE OF FONT	WITH clause/Output Returns a font.
Font selected (L)	WITH clause/Output Indicates whether a font has been selected.
Font string (A253)	WITH clause/Input/Output Input: preselected font, Output: font selected by end user.

### Example:

```
#FONT-NEW:= NULL-HANDLE /*Recommended every time you use this dialog
OPEN DIALOG NGU-FONT-SELECT
  USING #DLG$WINDOW
  GIVING #DLG
  WITH #FONT-NEW #FONT-SEL #FONT-STRING
```

# NGU-MESSAGEBOX Dialog

## Natural Object Name

NGUMBDI1

## Description

Displays a message box, for example with the title: "Dialog editor" and the message "Information message box" with an "OK" style push button that is selected by default. Please note that such a message box is modal (the end user can only leave it with a choice).

## OPEN DIALOG Parameters

Name/Data Type	Explanation
HANDLE OF GUI	USING clause/Input The parent dialog of the message box.
Button (A1)	WITH clause/Output Returns the selected button (for possible output values, see below).
Message (A253)	WITH clause/Input (BY VALUE) Here you specify the message to be issued.
Title (A50)	WITH clause/Input (BY VALUE) Here you specify the message box title.
Style (A32)	WITH clause/Input (BY VALUE) The type of message box (for possible input values, see below).

"Button" Value	Selected Button
<b>O</b>	OK push button.
<b>C</b>	Cancel push button.
<b>Y</b>	Yes push button.
<b>N</b>	No push button.
<b>R</b>	Retry push button.

"Style" Value	Message Box Type
<b>I</b>	blue lower-case "i" in a round icon, Information Provides information about the results of a command. Offers no user choices; the user acknowledges the message by clicking the OK button.
<b>!</b>	exclamation mark, Warning Alerts the user to a condition or situation that requires the user's decision and input before proceeding, such as an impending action with potentially destructive, irreversible consequences.
<b>S</b>	stop sign, Critical Informs the user of a serious problem that requires intervention or correction before work can continue.
<b>?</b>	This sign is no longer recommended as it does not clearly represent a type of message. The system continues to support its inclusion only for backward compatibility.
may be combined with:	
<b>O</b>	OK push button (default).
<b>OC</b>	OK and Cancel push buttons.
<b>YNC</b>	Yes, No, and Cancel push buttons.
<b>YN</b>	Yes and No push buttons.
<b>RC</b>	Retry and Cancel push buttons until the end user responds to the message box.
may be combined with:	
<b>1</b>	Make the first push button the default.
<b>2</b>	Make the second push button the default.
<b>3</b>	Make the third push button the default.

**Note:** If the messagebox has the style "C", an OK button is generated because a messagebox with only a Cancel button is not supported.

**Example:**

```

OPEN DIALOG NGU-MESSAGEBOX
  USING NULL-HANDLE
  WITH #BUTTON
  'Do you want to save the changes ?' /* These parameters are
  'Exit editor' '?YNC1'             /* passed BY VALUE
    
```

## Separator Keyword

### END-OF-LINE

You use this keyword to force line breaks in message box texts when using the NGU-MESSAGEBOX dialog. To do so, first embed this keyword in the alphanumeric "Message" string using the COMPRESS statement, for example:

```
COMPRESS 'HELLO' END-OF-LINE 'THIS IS A MESSAGE' TO #MESSAGE
```

You should, however, only use this technique if the text of the message clearly breaks into more than one line, because this overrides system-provided formatting.

## Usage in Enter or Leave Event Handlers

Before you open this dialog in an enter or leave event handler, you have to set the attributes SUPPRESS-ENTER-EVENT and SUPPRESS-LEAVE-EVENT to TRUE. This avoids an infinite loop on opening the dialog (when the messagebox appears, another enter and leave event occurs for the dialog or dialog element containing the enter or leave event handler). After the messagebox has been invoked, you modify the attributes to value FALSE again.

# NGU-SERVER-DATA Subprogram

## Natural Object Name

NGUSVDAT

## Description

Sends the data for service DDE-VIEW.SERVICE, topic DDE-VIEW.TOPIC, conversation-ID DDE-VIEW.CONV-ID, and item DDE-VIEW.ITEM to the client.

The data to be sent are contained in DDE-VIEW.DATA and must have been requested by the client in a "Request" or "Advise" message. DDE-VIEW.ITEM must be set to the item that was requested. Note that even if the "Advise" message resulted in a warm link, the data should still be specified. Natural will then decide whether the data will actually be sent to the client.

When sending data to the client, you must set DDE-VIEW.DATALEN to a value:

- If set to "0", the entire data will be sent; the client will send on return a value that indicates how many data were actually sent.
- If set to "-1", trailing blanks will be removed and the client will send on return a value that indicates how many data were actually sent.
- If set to a value greater zero, this amount of data is actually sent.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

# NGU-SERVER-GET-DATA Subprogram

## Natural Object Name

NGUSVGDA

## Description

Lets the server retrieve data supplied with a message from some client. You must use this subprogram in the event handler of a DDE-server event. DDE-VIEW.MESSAGE will hold one of the values described below. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE.

The possible values of DDE-VIEW.MESSAGE are:

<b>CONNECT</b>	On request by a client, a conversation has been established with the topic DDE-VIEW.TOPIC and the service DDE-VIEW.SERVICE. Topic and service must have been registered by the server. The value in DDE-VIEW.CONV-ID uniquely identifies this conversation.
<b>DISCONNECT</b>	On request by a client, the conversation identified by DDE-VIEW.CONV-ID has been closed.
<b>REQUEST</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested data as specified in DDE-VIEW.ITEM.
<b>ADVISE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested a data link for the data specified in DDE-VIEW.ITEM. The DDE-server event handler must then use the NGU-SERVER-DATA subprogram whenever new data become available during the conversation.
<b>UNADVISE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested to close a data link for the data specified in DDE-VIEW.ITEM. (This data link was previously opened using "Advise".)
<b>POKE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has sent data as specified in DDE-VIEW.ITEM and DDE-VIEW.DATA. These data have the length of DDE-VIEW.DATALEN. DDE-VIEW.DATA should be chosen as appropriate for the expected conversations.
<b>EXECUTE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested the execution of the command specified in DDE-VIEW.DATA and DDE-VIEW.ITEM. The length of this command is held in DDE-VIEW.DATALEN.
<b>TIMEOUT</b>	A timeout has occurred. No client message has been received during a specified time interval.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

The value in DDE-VIEW.TIMEOUT specifies the maximum number of milliseconds the server wants to wait for a message:

- If set to "-1", the server will wait indefinitely.
- If set to "0", a message will be returned immediately, if there is one pending, or "Timeout" will be returned.
- If set to any other value, the exact time that elapses before a message is returned can only be guaranteed to be no less than DDE-VIEW.TIMEOUT.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

# NGU-SERVER-WAIT Subprogram

## Natural Object Name

NGUSVWAT

## Description

Lets the server wait for data supplied with a message from some client. DDE-VIEW.MESSAGE will hold one of the values described below. You must then ensure that your event handler code acts on the value of DDE-VIEW.MESSAGE

The possible values of DDE-VIEW.MESSAGE are:

<b>CONNECT</b>	On request by a client, a conversation has been established with the topic DDE-VIEW.TOPIC and the service DDE-VIEW.SERVICE. Topic and service must have been registered by the server. The value in DDE-VIEW.CONV-ID uniquely identifies this conversation.
<b>DISCONNECT</b>	On request by a client, the conversation identified by DDE-VIEW.CONV-ID has been closed.
<b>REQUEST</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested data as specified in DDE-VIEW.ITEM.
<b>ADVISE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested a data link for the data specified in DDE-VIEW.ITEM. The DDE-server event handler must then use the NGU-SERVER-DATA subprogram whenever new data become available during the conversation.
<b>UNADVISE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested to close a data link for the data specified in DDE-VIEW.ITEM. (This data link was previously opened using "Advise".)
<b>POKE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has sent data as specified in DDE-VIEW.ITEM and DDE-VIEW.DATA. These data have the length of DDE-VIEW.DATALEN. DDE-VIEW.DATA should be chosen as appropriate for the expected conversations.
<b>EXECUTE</b>	The client of the conversation specified in DDE-VIEW.CONV-ID has requested the execution of the command specified in DDE-VIEW.DATA and DDE-VIEW.ITEM. The length of this command is held in DDE-VIEW.DATALEN.
<b>TIMEOUT</b>	A timeout has occurred. No client message has been received during a specified time interval.

When DDE-VIEW.MESSAGE has the value DISCONNECT, DATA, or NOTIFY, the fields DDE-VIEW.SERVICE and DDE-VIEW.TOPIC are also set appropriately.

The value in DDE-VIEW.TIMEOUT specifies the maximum number of milliseconds the server wants to wait for a message:

- If set to "-1", the server will wait indefinitely.
- If set to "0", a message will be returned immediately, if there is one pending, or "Timeout" will be returned.
- If set to any other value, the exact time that elapses before a message is returned can only be guaranteed to be no less than DDE-VIEW.TIMEOUT.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

# NGU-SERVER-REGISTER Subprogram

## Natural Object Name

NGUSVREG

## Description

This subprogram makes it known that the server supports the service specified in DDE-VIEW.SERVICE and the conversations on the topic specified in DDE-VIEW.TOPIC. A client will not be able to establish a conversation with the server until this subprogram is executed. Your client should not try to register the name "Natural" because this is reserved.

Before you execute this subprogram, you must assign the \*DIALOG-ID system variable of the server dialog to DDE-VIEW.CONV-ID. You must also assign the string "DLGID" to DDE-VIEW.MESSAGE.

## Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-SERVER-STOP Subprogram

### Natural Object Name

NGUSVSTP

### Description

Terminates all DDE activity for the server. Any topics registered for the service are unregistered and any open conversations are terminated. The server should not call any other DDE transaction after this subprogram has been executed.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

## NGU-SERVER-UNREGISTER Subprogram

### Natural Object Name

NGUSVUNR

### Description

Makes the topic DDE-VIEW.TOPIC unavailable. Any open conversations on that topic should be closed first.

### Parameters

Name/Data Type	Explanation
1 DDE-VIEW	Input/Output PDA for DDE subprograms.
2 SERVICE (A20)	Service name.
2 TOPIC (A20)	Topic name.
2 CONV-ID (I4)	Conversation ID.
2 MESSAGE (A20)	Message information.
2 ITEM (A20)	Item of current conversation.
2 FORMAT (A20)	Format of data to be sent.
2 TIMEOUT (I4)	Time interval before the conversation is interrupted because there is no server message.
2 DATALEN (I2)	Length of data.
2 DATA-ARRAY (A1/1:V)	
1 Max-Index (I2)	Input Highest index of DATA-ARRAY; can be set to any I2 value.
1 Response (I4)	Output Natural error (if applicable).

# Enhanced Source Code Format

The dialog editor supports the following source formats:

- 213. This is the format generated by Natural Version 2.1.3 (New Dimension). It is supported for input only. You cannot generate 2.1.3 format with Natural Version 2.2 and with Natural Version 3.
- 22C. This is the format generated by Natural Version 2.2.2. As from Natural for Windows and Unix/OpenVMS Version 4.1, this format cannot be generated.
- 22D. This is the standard "enhanced" source code format. It is generated for compiling, storing, and editing dialogs in Natural Version 2.2.3 and above.

This section describes the syntax conventions for entering code in the source code window provided by the program editor.

- Syntax Conventions
- How Natural Dialogs Work
- Syntax

## Syntax Conventions

Syntax is described using the following meta-notation:

*syntax\_element\_name ::= description*

Syntax Element	Represents
<i>syntax_element_name ::=</i>	Identifies the construct whose structure is defined by <i>description</i> .
<i>description</i>	Program code displayed in UPPER-CASE letters.

The syntax symbols used in this section are explained in the Natural Statements documentation. In addition, the following symbols are used within the diagrams:

Symbol	Represents
{ <i>element1</i>   <i>element2</i> }	Elements contained within braces and separated by vertical bars indicate that exactly one of the elements must be specified.
< >	Pairs of angle brackets indicate that code is separated into multiple lines. Each pair of brackets represents the end of a Natural source line.
/*[ and /*]	These bracket pairs form a collapsible block in list mode, even in user code sections.

Line-number references are explicitly *not* supported within dialogs, as the dialog editor will generate line numbers for dialog sources without consideration of any line-number references.

Line length and source size are subject to the general limits imposed by Natural.

**Note:**

Where syntax elements are shown separated by blanks, blanks are required, but the exact number of blanks is not significant.

## How Natural Dialogs Work

This section gives you some background information so you can better understand what the various elements of the dialog source do.

A Natural dialog is an executable module. You invoke it either directly from the Natural environment or from another executing module with an OPEN DIALOG statement. Invoking the dialog causes the dialog's executable to be instantiated and executed with the system variables \*CONTROL equal to NULL-HANDLE and \*EVENT equal to "OPEN". This OPEN event is processed by calling the inline subroutine \*DLG\$\$SUBR\$CREATE\$WINDOW which creates the dialog window. Window creation invokes the dialog with \*CONTROL equal to NULL-HANDLE and \*EVENT equal to "AFTER-OPEN", to which the dialog responds by calling the inline subroutine \*DLG\$\$SUBR\$CREATE\$CONTROLS. This subroutine creates any dialog elements defined for the dialog, and then calls PROCESS GUI ACTION AFTER-CREATION to notify Natural that window creation is complete.

Depending on further application processing and user input, the dialog will repeatedly be executed with \*CONTROL and \*EVENT set appropriately. The dialog remains instantiated until it is explicitly unloaded as a result of the window being destroyed. This is the default reaction to the CLOSE event which in turn occurs as a result of either end-user interaction or the CLOSE DIALOG statement.

The same dialog module may be instantiated more than once, resulting in more than one dialog and its window and dialog elements being active. Each dialog has a unique numeric ID which is available as \*DIALOG-ID during execution of that dialog, and as *window\_handle.CLIENT-DATA* wherever that dialog's window handle is accessible (the window must be created with that attribute setting).

## Syntax

- General Syntax
- Subsections of the General Syntax
- Subordinate Syntax Sections Often Used in the Syntax Diagrams

### General Syntax

This is the complete source of a dialog.

```
dialog_source_22D
 ::=

  /** DIALOG SOURCE 22D dialog_info_section_22D
  [dialog_options_section_22D]
  dialog_data_section_22D [user_subroutines_section_22D]
  window_definition_22D control_definitions_22D
  default_handler_section_22D
  error_handler_section_22D before_any_section_22D
  event_handlers_22D after_any_section_22D
  END /** END-DIALOG-SOURCE
```

## Subsections of the General Syntax

### dialog\_info\_section\_22D

```
dialog_info_section_22D
:=:
```

- \* The dialog's "info" section, consisting of a banner line, frame gallery information, and an optional comment.

```
/*[ DEFINE DIALOG INFO
```

- \* The following line is always generated by the dialog editor, but ignored on input.

```
/*D* Natural Dialog Description version_string / date
time
```

- \* One or more following lines are present in dialogs generated using the frame gallery.
- \* If present, they are preserved by (but cannot be changed in) the dialog editor.

```
[/*DF frame_gallery_info]...
```

- \* The empty comment line is generated by the dialog editor if no user comment is present, but ignored on input.

```
[dialog_comment ::= {/** EMPTY DIALOG COMMENT | [user_code_line_protected_by_/**_prefix]...}]
/*] END-DIALOG-INFO
```

### dialog\_options\_section\_22D

```
dialog_options_section_22D
:=:
```

- \* These option settings are only generated by the dialog editor if the "save settings with dialog" option is on. When a dialog is being loaded and this section is present, the option is turned on.

```
/*[ DEFINE OPTION SETTINGS {/** SET option_name option_value}...
/*] END-OPTION-SETTINGS]
```

**dialog\_data\_section\_22D**

```
dialog_data_section_22D  
:=:
```

- \* The dialog's data section, consisting of a global, parameter, and local data area.
- \* The sequence is fixed.

```
DEFINE DATA gda_section_22D  
pda_section_22D lda_section_22D  
END-DEFINE
```

**Subsections of the dialog\_data\_section\_22D**

*gda\_section\_22D*

```
gda_section_22D
:=:
```

\* The optional global data area specification. Note that it may reference blocks.

```
/*[ DEFINE GLOBAL DATA [GLOBAL USING gda_specification] /*] END-GLOBAL-DATA
```

*pda\_section\_22D*

```
pda_section_22D
:=:
```

\* The dialog's parameter data area must always contain the parent handle as the  
\* first field.

```
/*[ DEFINE DIALOG PARAMETERS PARAMETER 01 #DLG$PARENT HANDLE OF
GUI BY VALUE user_data_section_22D
/*[ DEFINE USING [PARAMETER USING pda_name]... /*] END-USING /*] END-DIALOG-PARAMETERS
```

*lda\_section\_22D*

```
lda_section_22D
:=:
```

\* The dialog's local data area. The handle declarations for all dialog elements are  
\* necessary if the dialog is to be compiled, but are ignored by the dialog editor  
\* on input and re-generated from the actual dialog element definitions below.

```
/*[ DEFINE LOCAL DATA LOCAL /*[ DEFINE HANDLES {01 control_name
HANDLE OF control_class[<>]}... /*] END-HANDLES user_data_section_22D
/*[ DEFINE USING [LOCAL USING lda_name]... /*] END-USING /*] END-LOCAL-DATA
```

**user\_subroutines\_section\_22D**

```
user_subroutines_section_22D
:=:
```

\* The list of user-defined subroutines. The enclosing pseudo-syntax is generated  
\* only if there actually are subroutines.

```
/*[ DEFINE SUBROUTINES [DEFINE SUBROUTINE subroutine_name
user_code_section_22D END-SUBROUTINE]...
/*] END-SUBROUTINES
```

**window\_definition\_22D**

```
window_definition_22D
:=:
```

\* The window is defined within a standard subroutine (this must always be present).

```

DEFINE SUBROUTINE #DLG$ SUBR$CREATE$WINDOW /** DEFINE CONTROL window_name
  non_array_control_definition_22D
  END-SUBROUTINE /** END-CONTROL

```

## control\_definitions\_22D

```

control_definitions_22D
:=:

```

- \* All dialog elements are defined within one standard subroutine (this must always be present). The sequence generated by the dialog editor is: menu bar,
- \* tool bar, font controls, timers, any other dialog elements.

```

DEFINE SUBROUTINE #DLG$SUBR$CREATE$CONTROLS /** DEFINE DIALOG
ELEMENTS [control_definition :=:

```

- \* Each dialog element definition is enclosed in pseudo-comments.

```

/*[
  DEFINE CONTROL control_name[array_bounds]

```

- \* The following optional comment appears in the source code only.

```

[control_comment
  :=: {user_code_line_protected_by_/**_prefix}...]

```

- \* The following code creates the dialog element.

```

{non_array_control_definition_22D
  | array_control_definition_22D}

```

- \* List box controls and selection-box controls may have items lists defined as one array of dialog elements. These follow the dialog element creation code.

```

[control_items :=: /* DEFINE ITEMS control_name_array_bounds
  array_control_definition_22D
  /*] END-ITEMS ] /*] END-CONTROL

```

```

]...
END-SUBROUTINE /** END-DIALOG-ELEMENTS

```

## default\_handler\_section\_22D

```

default_handler_section_22D
:=:

```

- \* The DEFAULT event handler, specified as subroutine. For compilation, it must be present, as this subroutine is called from various places.

```

DEFINE SUBROUTINE
  #DLG$HANDLER$DEFAULT /** DEFINE EVENT DEFAULT user_code_section_22D
  END-SUBROUTINE /** END-EVENT

```

**error\_handler\_section\_22D**

```
error_handler_section_22D
:::
```

\* The ERROR event handler, specified as ON ERROR section. Optional.

```
ON ERROR /** DEFINE
  EVENT ERROR user_code_section_22D
END-ERROR /** END-EVENT
```

**before\_any\_section\_22D**

```
before_any_section_22D
:::
```

\* The BEFORE-ANY event handler, that is, the code which precedes the DECIDE  
\* statements which evaluate \*CONTROL and \*EVENT.

```
/*[ DEFINE EVENT BEFORE-ANY user_code_section_22D
  /*] END-EVENT
```

**event\_handlers\_22D**

```
event_handlers_22D
:::
```

\* The DECIDE statements which evaluate first \*CONTROL, then \*EVENT,  
\* activating the appropriate event handlers.

```
DECIDE
ON FIRST *CONTROL /** DEFINE ALL EVENTS {dialog_events ::= /*[ DEFINE
EVENTS FOR DIALOG VALUE NULL-HANDLE
```

\* \*CONTROL = NULL-HANDLE indicates an event associated either with the  
\* window or with the dialog itself.

```
DECIDE ON FIRST *EVENT /*[
  DEFINE EVENT OPEN VALUE 'OPEN'
```

\* The OPEN event handler starts with any BEFORE-OPEN user code and ends  
\* with the window creation call.

```
user_code_section_22D
  PERFORM #DLG$SUBR$CREATE$WINDOW /*] END-EVENT /*[ DEFINE EVENT AFTER-OPEN VALUE
  'AFTER-OPEN'
```

\* The AFTER-OPEN event occurs while the window creation call is being  
\* processed, that is, it is a nested call of the dialog. It includes the creation of all  
\* dialog elements and the assignment of those window attributes which use  
\* dialog element handles (for example, the handle of the menu bar). User  
\* code may follow.

```
PERFORM #DLG$SUBR$CREATE$CONTROLS [extra_window_attributes_22D]
  PROCESS GUI ACTION AFTER-CREATION WITH window_name PROCESS GUI ACTION
  RESET-ATTRIBUTES
```

\* The user's after-open code follows.

```
user_code_section_22D
  /*] END-EVENT /*[ DEFINE EVENT CLOSE
```

\* The CLOSE event occurs either because the end user directly closes the  
\* window, or because the parent window is closed, or as the result of a CLOSE  
\* DIALOG statement.

```
VALUE 'CLOSE'
  {dialog_close_handler_22D ::= user_code_section_22D}
```

\* The following call "destroys" the window and unloads the dialog.

```
PROCESS GUI ACTION DELETE-WINDOW WITH window_name
```

\* The following statement leaves the dialog, bypassing the AFTER-ANY  
\* handler.

```
ESCAPE ROUTINE IMMEDIATE /*] END-EVENT
```

\* The following dialog event handlers are optional and include any user-defined  
\* events.

```
[event_handler_section_22D...]
  NONE PERFORM #DLG$HANDLER$DEFAULT END-DECIDE /*] END-EVENTS
```

```
}
```

\* All event handlers for dialog elements defined in  
\* \*DLG\$SUBR\$CREATE\$CONTROLS are listed below.

```
[control_events_section_22D...]
  NONE PERFORM #DLG$HANDLER$DEFAULT END-DECIDE /** END-ALL-EVENTS
```

\* All events that are not processed and have not been suppressed are handled in the  
\* DEFAULT event handler. Any dialog elements created outside  
\* \*DLG\$SUBR\$CREATE\$CONTROLS are handled there.

### **after\_any\_section\_22D**

```
after_any_section_22D
  ::=
```

\* The AFTER-ANY event handler, that is, the code which follows the DECIDE  
\* statements evaluating \*CONTROL and \*EVENT.

```
/*[ DEFINE EVENT AFTER-ANY user_code_section_22D
  /*] END-EVENT
```

## Subordinate Syntax Sections

### user\_data\_section\_22D

```
user_data_section_22D
:::
```

- \* A section with data declarations. The dialog editor generates a comment if no user
- \* code is present. The user code layout is preserved, except that the whole section is
- \* indented appropriately.

```
[frame_code_section_22D]
{/** EMPTY USER CODE SECTION | user_code_line_22D...}
[frame_code_section_22D]
```

### user\_code\_section\_22D

```
user_code_section_22D
:::
```

- \* A section with executable statements. The dialog editor generates a comment if
- \* no user code is present. The user code layout is preserved, except that the whole
- \* section is indented appropriately.

```
[frame_code_section_22D]
{/** EMPTY USER CODE SECTION | user_code_line_22D...}
[frame_code_section_22D]
```

### frame\_code\_section\_22D

```
frame_code_section_22D
:::
```

- \* This protected section is code in a frame gallery dialog. Do not change this code.

```
/*[ DEFINE FRAME CODE [user_code_line_22D...]
/*] END-FRAME-CODE
```

### user\_code\_line\_22D

```
user_code_line_22D
:::
```

- \* A code line within an event section or subroutine. Indentation is preserved by the
- \* dialog editor. However, a minimum indentation is enforced.

```
indented_code_line
```

### event\_handler\_section\_22D

```
event_handler_section_22D
:::
```

- \* A single event handler section, that is, the VALUE clause for the event in a DECIDE
- \* statement for the relevant dialog element.

```

/*[ DEFINE EVENT event_name
  VALUE 'event_name' user_code_section_22D
  /*] END-EVENT

```

### control\_events\_section\_22D

```

control_events_section_22D
:=:

```

- \* The collection of all event handlers for one dialog element, that is, a VALUE clause
- \* for the dialog element's handle containing a DECIDE statement for \*EVENT. If the
- \* dialog element is an array, all elements are handled in this section.

```

/*[
  DEFINE EVENTS FOR control_name VALUE control_name[( *[, *)] DECIDE
  ON FIRST *EVENT event_handler_section_22D...
  NONE PERFORM #DLG$HANDLER$DEFAULT END-DECIDE /*] END-EVENTS

```

### non\_array\_control\_definition\_22D

```

non_array_control_definition_22D
:=:

```

- \* A non-array dialog element definition corresponds to the PROCESS GUI statement
- \* action that creates the dialog element. As the WITH PARAMETERS ...
- \* END-PARAMETERS clause is used, all attributes not mentioned have their default
- \* values. The dialog editor does not generate such attributes.
- \* The first and the second attributes must be HANDLE-VARIABLE and TYPE.
- \* The dialog editor uses the predefined attribute value names defined
- \* in the standard local data area NGULKEY1 if this data area has been included in
- \* the dialog. If it is not included, the dialog cannot be compiled. Note that variable
- \* references are allowed for only a subset of the attributes.

```

PROCESS GUI ACTION ADD WITH
  PARAMETERS {attribute_name = {constant | variable_reference}<>}...
  END-PARAMETERS [frame_code_line_22D...] /*] END-CONTROL

```

### array\_control\_definition\_22D

```

array_control_definition_22D
:=:

```

- \* An array definition of a dialog element consists of one PROCESS GUI statement for
- \* each array element. Only a subset of dialog elements may be defined as arrays (for
- \* example, not the dialog window or list box controls). Instead of the WITH
- \* PARAMETERS ... END-PARAMETERS clause, explicit attribute assignments and
- \* the simple WITH clause are used, as array elements will share many equal but
- \* non-default attribute values.
- \* The dialog editor is quite restrictive with respect to dialog element arrays:
- \* a maximum of two dimensions is allowed; elements cannot be placed individually;
- \* only a subset of the attributes may vary between elements.
- \* The dialog editor will, when scanning dialog element array definitions, take those
- \* attribute values which may not vary from the first element's definition. Specifically,
- \* all coordinate definitions but the first will be derived from the pseudo-attributes
- \* H-SPACING, V-SPACING, and ARRANGE-IN-COLUMNS. (Note, however,
- \* that not all dialog elements have coordinates.) All attributes not mentioned explicitly
- \* have their default values; this is enforced by the initial PROCESS GUI ACTION

\* RESET-ALL.

```
PROCESS GUI ACTION RESET-ATTRIBUTES [spacing_info:::
  /** H-SPACING = horiz_spacing /** V-SPACING = vert_spacing /**
  ARRANGE-IN-COLUMNS = {TRUE|FALSE}<>] ] {
```

- \* Creation of one array element. Numeric references to a message file string must
- \* be handled in an explicit PROCESS GUI statement action for that string, as
- \* direct assignment to a number would simply convert that number to a string.

```
[attribute_assignment::: {
  control_name_and_index.attribute_name := attribute_value |
  PROCESS GUI ACTION GET-MESSAGE-TEXT WITH number indexed_control_name
  }<>]... PROCESS GUI ACTION ADD WITH parent_name type_name control_name_and_index
}...
```

### extra\_control\_attributes\_22D

```
extra_control_attributes_22D
:=:
```

- \* Some attributes cannot be defined meaningfully in the PROCESS GUI ACTION
- \* ADD that creates the dialog element; an example is a non-modifiable selection-box
- \* control's STRING, which must be one of the items' STRING values. This attribute
- \* is therefore defined after all items have been added.

```
control_name_and_index.attribute_name
:= attribute_value
```

### extra\_window\_attributes\_22D

```
extra_window_attributes_22D
:=:
```

- \* Some attributes cannot be defined meaningfully in the PROCESS GUI ACTION
- \* ADD that creates the dialog, for example, the default button can only
- \* be defined after that button has been created.

```
window_name.attribute_name
:= attribute_value
```