

Component Browser

The following topics are covered below:

- Introduction
 - User Interface
 - Application Development Support
-

Introduction

The Component Browser can be used to view ActiveX components which are available for developing NaturalX applications. It presents the information in a way that is especially useful to Natural application developers.

The Component Browser comprises the following features:

- Available ActiveX components and their dispatch and dual interfaces are listed.
- Data types are mapped to Natural formats.
- The external components' help files are directly accessible.
- Natural programming examples are automatically generated.
- Many programming errors can be prevented.

User Interface

- Tree View
- Data View
- Interaction Tree View and Data View
- Menu

The Component Browser uses a split window. The left pane shows a tree view representing the available external components and the right pane shows information on a selected item.

Tree View

Groups

At startup the Component Browser's tree view consists of four nodes that group the available external components.

All ActiveX Components

This group lists ActiveX Controls and Automation Objects.

ActiveX Controls

This group lists only the ActiveX Controls.

Automation Objects

This group lists the Automation Objects.

Interfaces

The last group lists all dual and dispatch interfaces that can be addressed in a Natural application. In this context, their relation to an ActiveX component is not taken into account.

Tree Nodes

In general, a node in the tree view represents either an ActiveX component or an interface. It provides textual information about the node and has a specific icon assigned that represents additional information.

The following table lists all available nodes with their icons and gives a short description:

Type	Icon	Description
Group		Group node.
ActiveX component		ActiveX component.
Interface		Interface of the current ActiveX component.
Default interface		Default interface of the current ActiveX component.

Order

By default, the ActiveX component nodes are inserted in alphabetical order of their external names. If you wish to see them sorted according to their ProgID, select menu 'View > Show by ProgID'.

Interface nodes are always inserted in alphabetical order.

Data View

The data view uses a property sheet to display the specific information for a selected node. This sheet consists of four pages: **General**, **Properties**, **Events** and **Methods**.

General

Components and interface specific information such as name, Global Unique ID and help file name. It is always the active page if a new node is selected.

Properties

Specific information about the properties offered by an interface. These are, for example, the property's name and type.

Events

Specific information on a component's event interface. These are, for example, the event's name and parameters.

Methods

Specific information about the methods and GET properties and PUT properties offered by an interface. Displayed are, for example, the method's name, return type and parameters.

These pages are discussed in more detail in the context of interaction between tree and data view.

Interaction Tree View and Data View

The contents of the 'data view' depend on the object selected in the tree view.

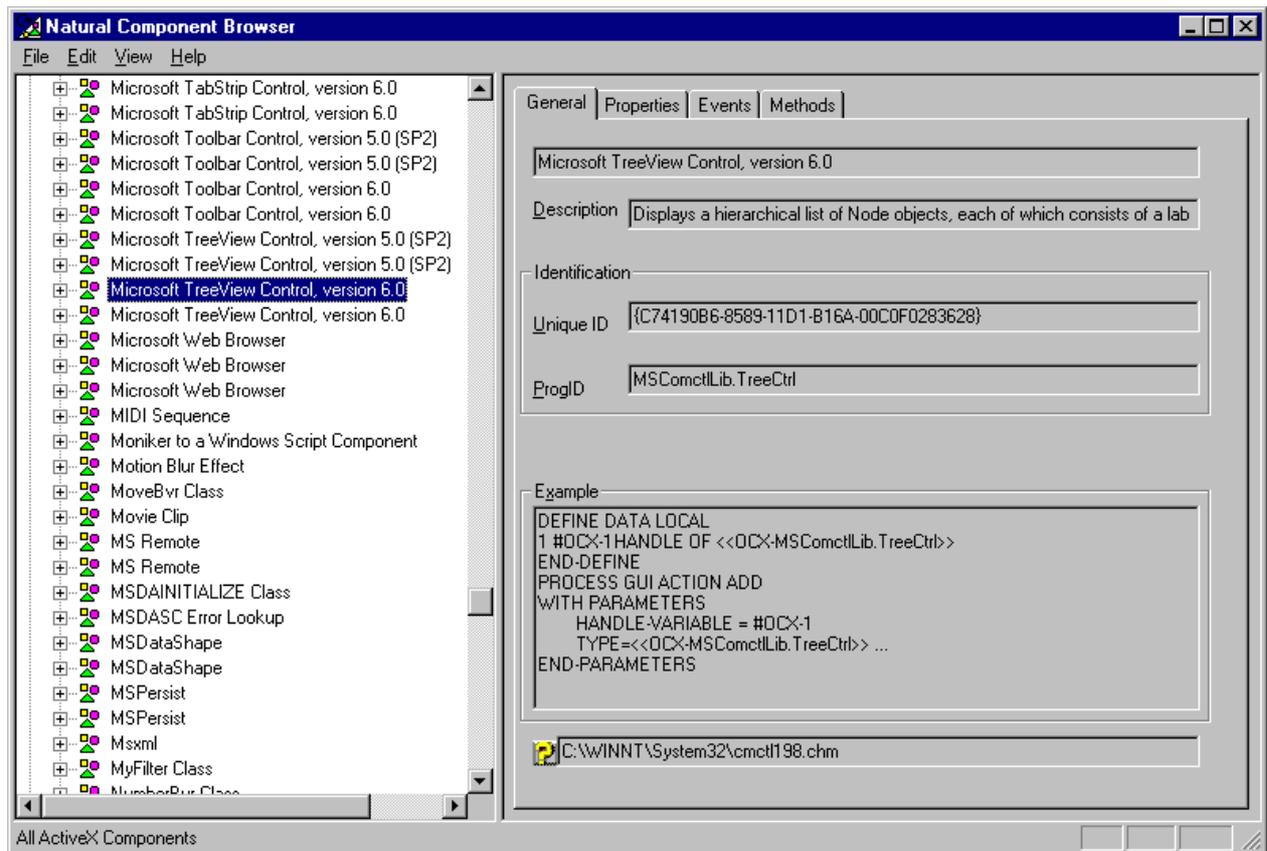
Group

If any of the group nodes 'All ActiveX Components', 'ActiveX Controls', 'Automation Objects' or 'Interfaces' is selected, the data view remains empty.

ActiveX Component

If a component node is selected, all four property pages are available. The page General provides the following information:

Name	Component name
Description	Short textual description.
Unique ID	Global Unique ID (GUID).
ProgID	ProgID.
Example	Natural statements that show how to use the component. (see Example Construction)
Help	Help file name. This file can be opened by pressing 



If a component is selected and the page **Properties**, **Events** or **Methods** is activated, the information displayed on this page refers to the default interface.

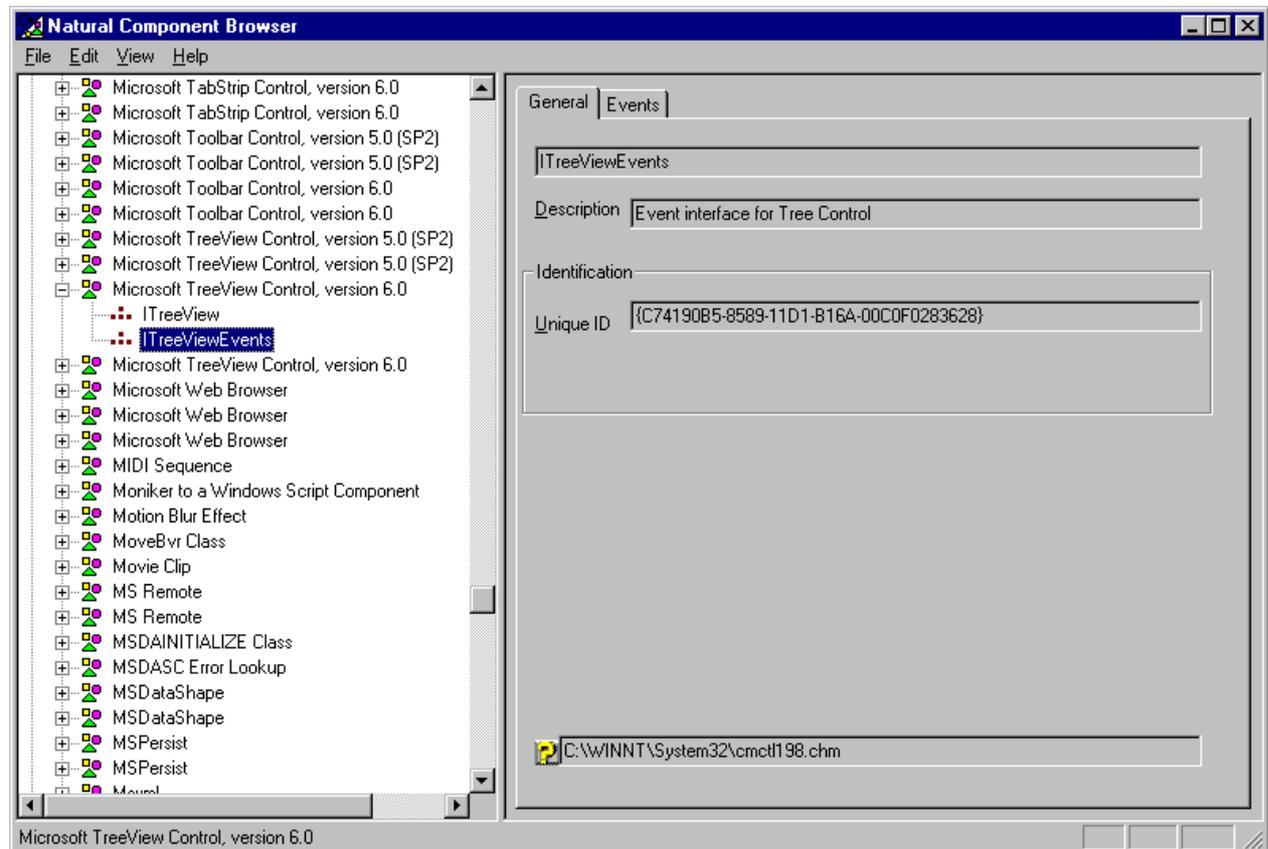
Interface

If an interface node is selected the number of available property pages depends on the type of the interface.

- If the Interface is browsed in the context of a component and if it is an Event Interface, then only the pages **General** and **Events** are set.
- Otherwise, the pages **General**, **Properties** and **Methods** are set.

In both cases, the page **General** provides the following information:

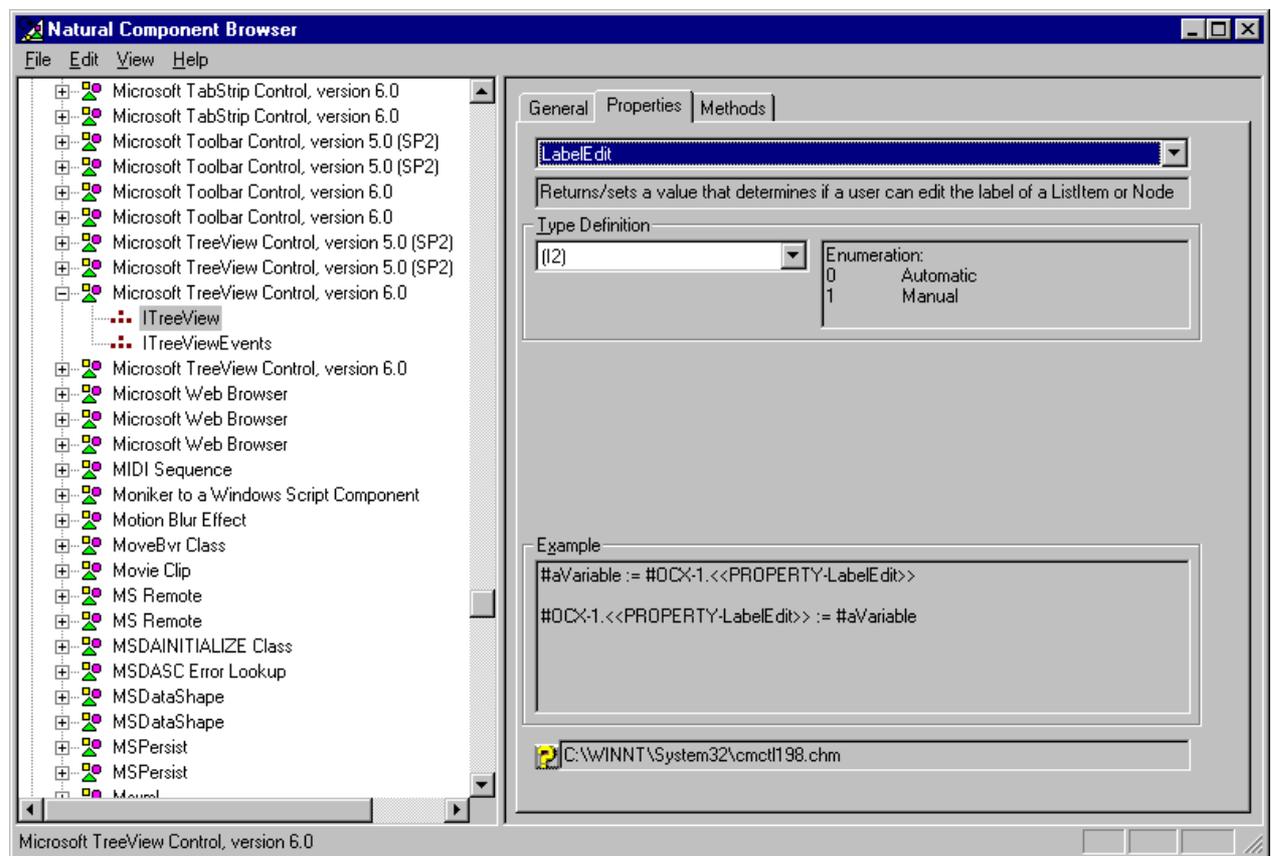
Name	Interface name
Description	Short textual description.
Unique ID	Global Unique ID (GUID).
Help	Help file name. This file can be opened by pressing 



Interfaces

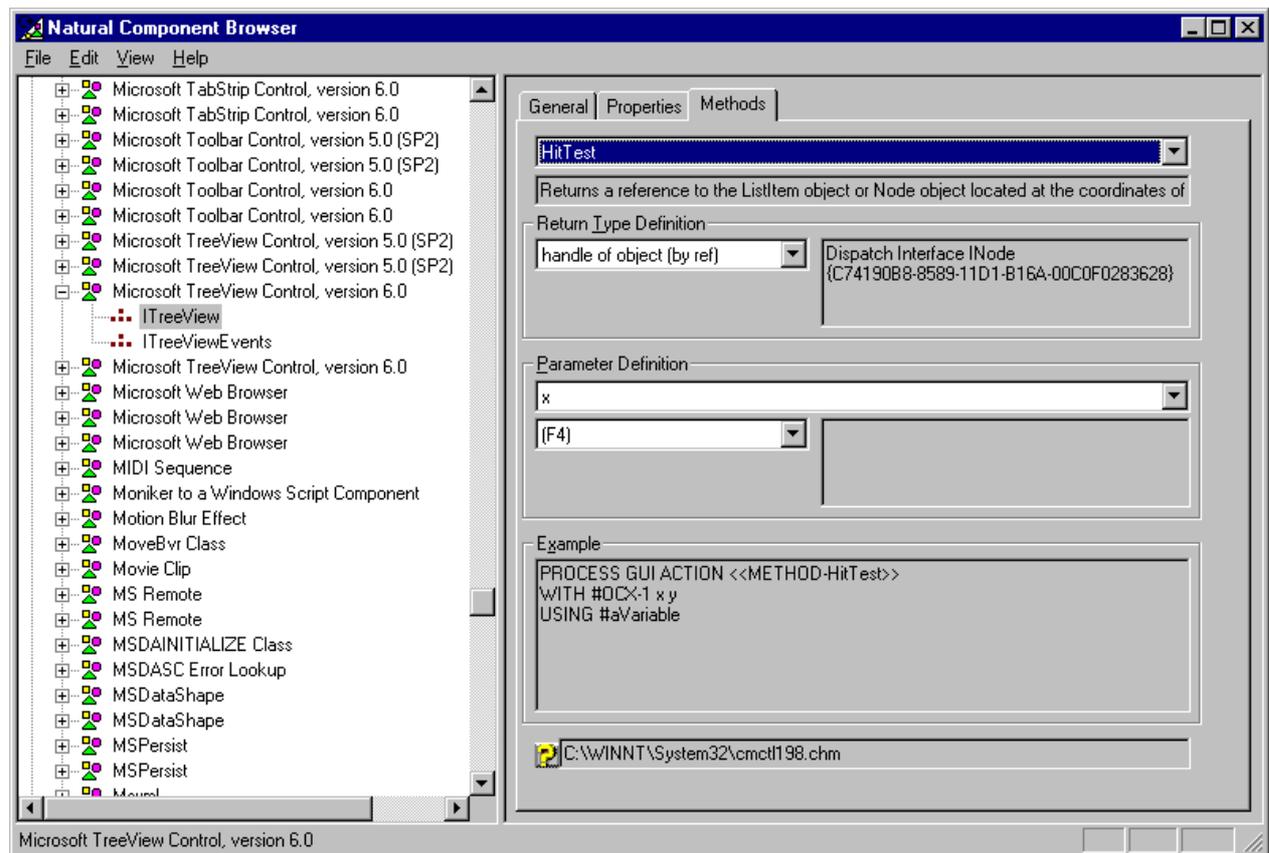
The page **Properties** provides the list of properties that belong to the selected interface. For a specific property the following information is displayed:

Description	Short textual description for the selected property.
Type Definition	List of valid Natural formats for the property's type. Additional type info on the selected Natural format, e.g. valid values for enumeration types.
Example	(see Example Construction).
Help	Help file name. This file can be opened by pressing 



The page **Methods** provides the list of methods that belong to the selected interface. This list includes GET-properties and PUT-properties. For a specific method or GET or PUT property the following information is displayed:

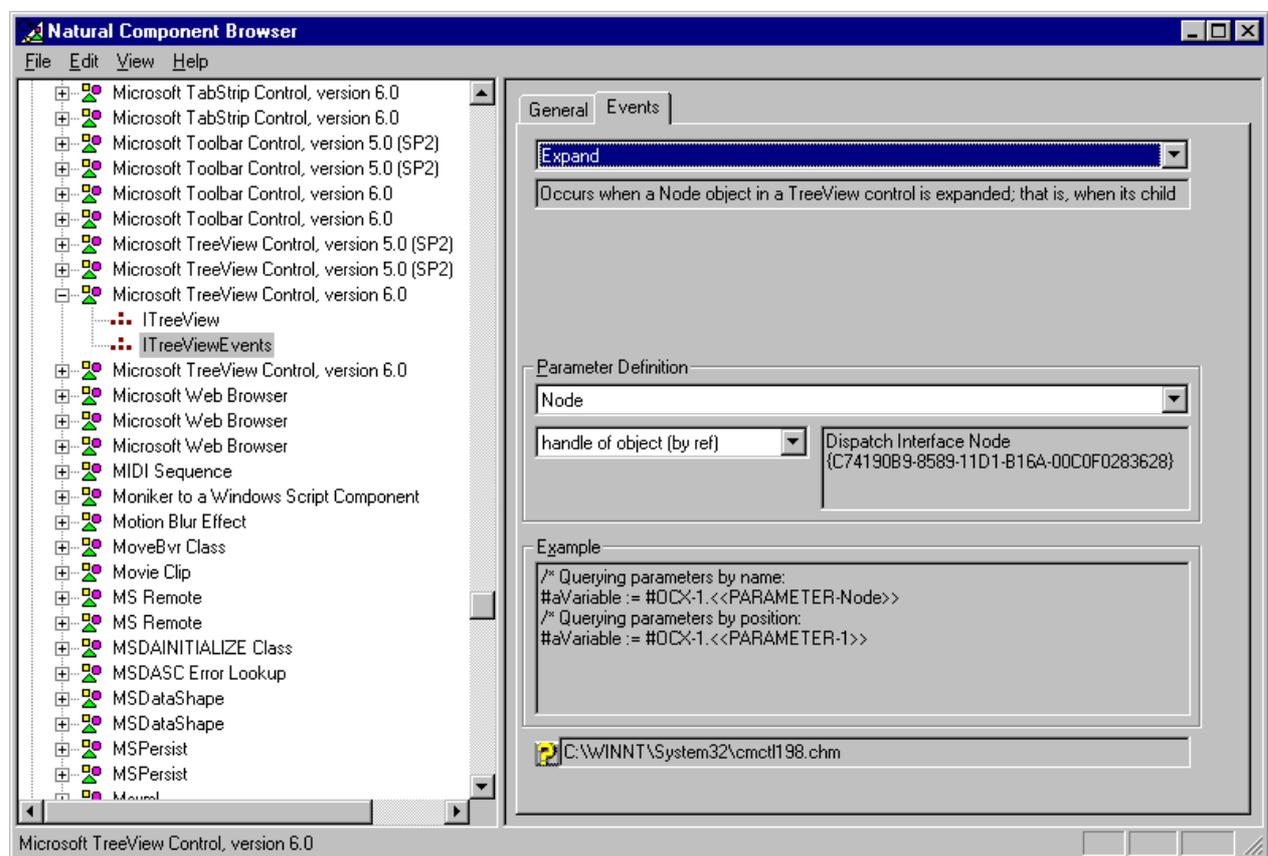
Description	Short textual description for the selected method.
Return Type Definition	List of valid Natural formats for the method's type. Additional type info on a selected Natural format, e.g. Name and GUID of the interface if a handle of object corresponds to a dispatch interface. This interface can then be found in group 'Interfaces'.
Parameter Definition	List of parameters that are required by the method. List of Natural formats for each parameter together with additional info on the call mode (by ref). Additional type info on a selected Natural format, e.g. Name and GUID of the interface if a handle of object corresponds to a dispatch interface. This interface can then be found in group 'Interfaces'.
Example	(see Example Construction).
Help	Help file name. This file can be opened by pressing 



Event Interfaces

The page **Events** provides the list of events that belong to the selected interface. For a specific event the following information is displayed:

Description	Short textual description for the selected event.
Parameter Definition	List of parameters that are required by the event. List of valid Natural formats for each parameter together with additional info on the call mode (by ref). Additional type info on selected Natural format, e.g. Name and GUID of the interface if a handle of object corresponds to a dispatch interface. This interface can then be found in group 'Interfaces'.
Example	(see Example Construction).
Help	Help file name. This file can be opened by pressing 



Menu

File

Item	Description
EXIT	Leaves the Component Browser.

Edit

Item	Description
Copy	This item is enabled if the data view has the focus. The selected text is copied to the clipboard.
Copy CLSID to Clipboard	This item is enabled if either the tree view or the data view has the focus. A component's CLSID is copied to the clipboard.
Copy ProgID to Clipboard	This item is enabled if either the tree view or the data view has the focus. A component's ProgID is copied to the clipboard.

View

Item	Description
Show by ProgID	This item is enabled if the tree view has the focus. By default, the tree view is sorted according to the component's external names. If this option is checked, it is sorted according to their ProgIDs. The currently displayed information is updated.
Show Current Version	This item is enabled if the tree view has the focus. By default, the tree view shows all versions of a component. If this option is checked, only the current version of a component is shown. The currently displayed information is updated.
Status Bar	Shows or hides the status bar.
Refresh	This item is enabled if the tree view has the focus. The tree view is refreshed, i.e. the information currently displayed is updated.

Help

Item	Description
About Component Browser	This item is enabled if either the tree view or the data view has the focus. Information on copyrights, version etc. is displayed.

Application Development Support

Example Construction

The data view provides detailed examples of Natural code that show how to use a selected object in a Natural application. Which statements are generated depends on the object's type.

The example code or just parts of it can be selected, copied to the clipboard and used directly in an application. Only variable names might have to be adapted to meet the application's requirements.

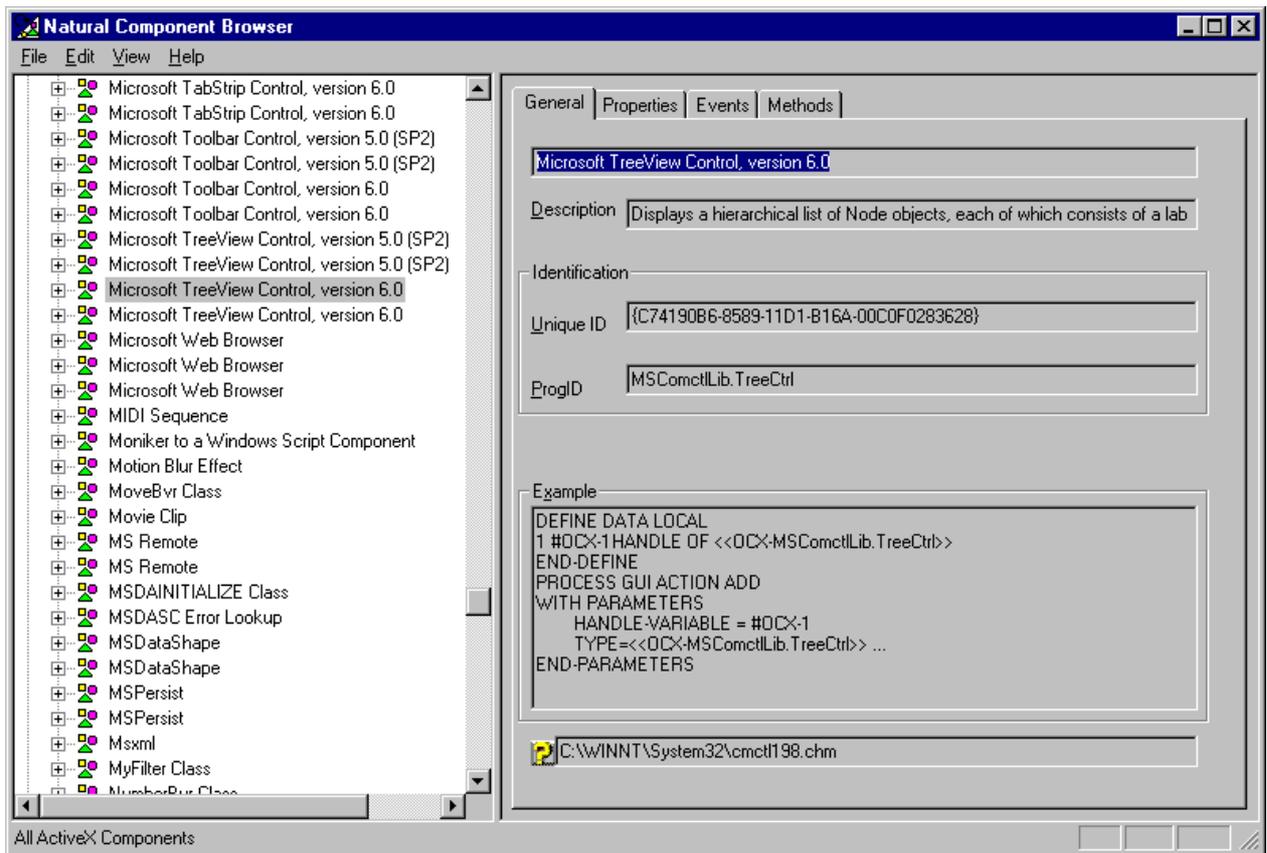
Frequently used identifiers such as CLSID and ProgID can be copied directly to the clipboard using menu item 'Edit > Copy CLSID to Clipboard' or 'Edit > Copy ProgID to Clipboard'.

ActiveX Controls

For ActiveX controls, the appropriate PROCESS GUI statement is generated that shows how to use these components in Natural applications.

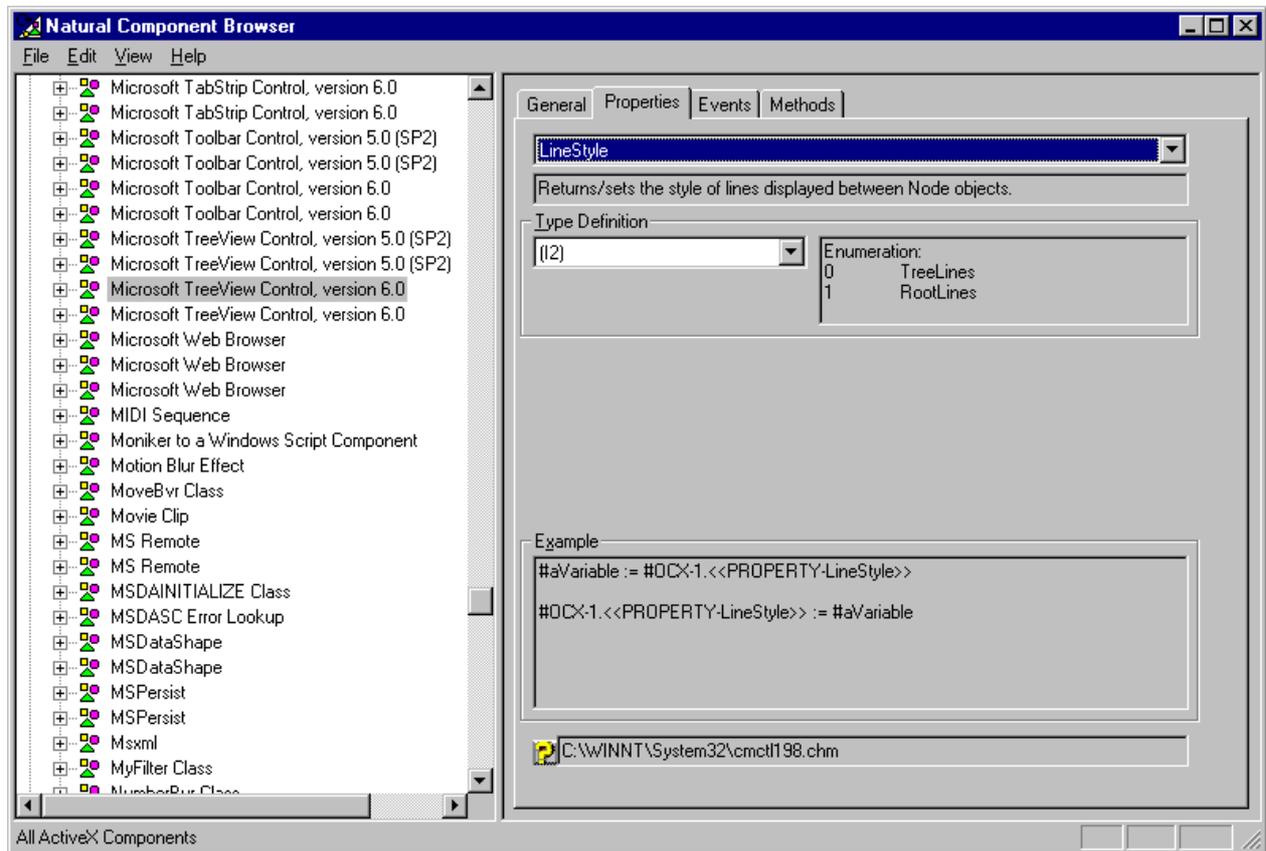
General

The example on the page **General** shows how an object of this type is instantiated. Here #OCX-1 denotes a variable that can be adapted to the current application.



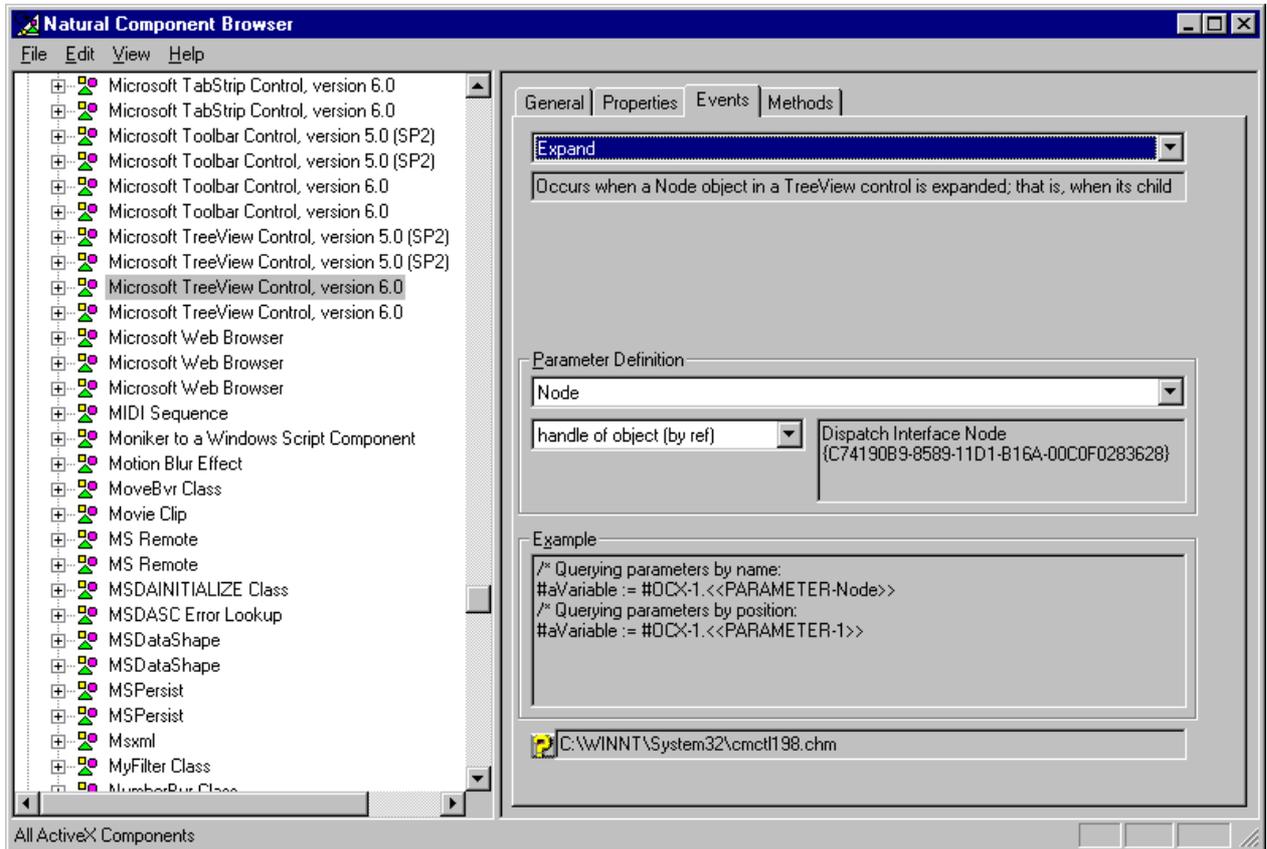
Properties

The example on the page **Properties** shows how to assign a property to a variable and how to assign a variable to a property. Here #OCX-1 is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.



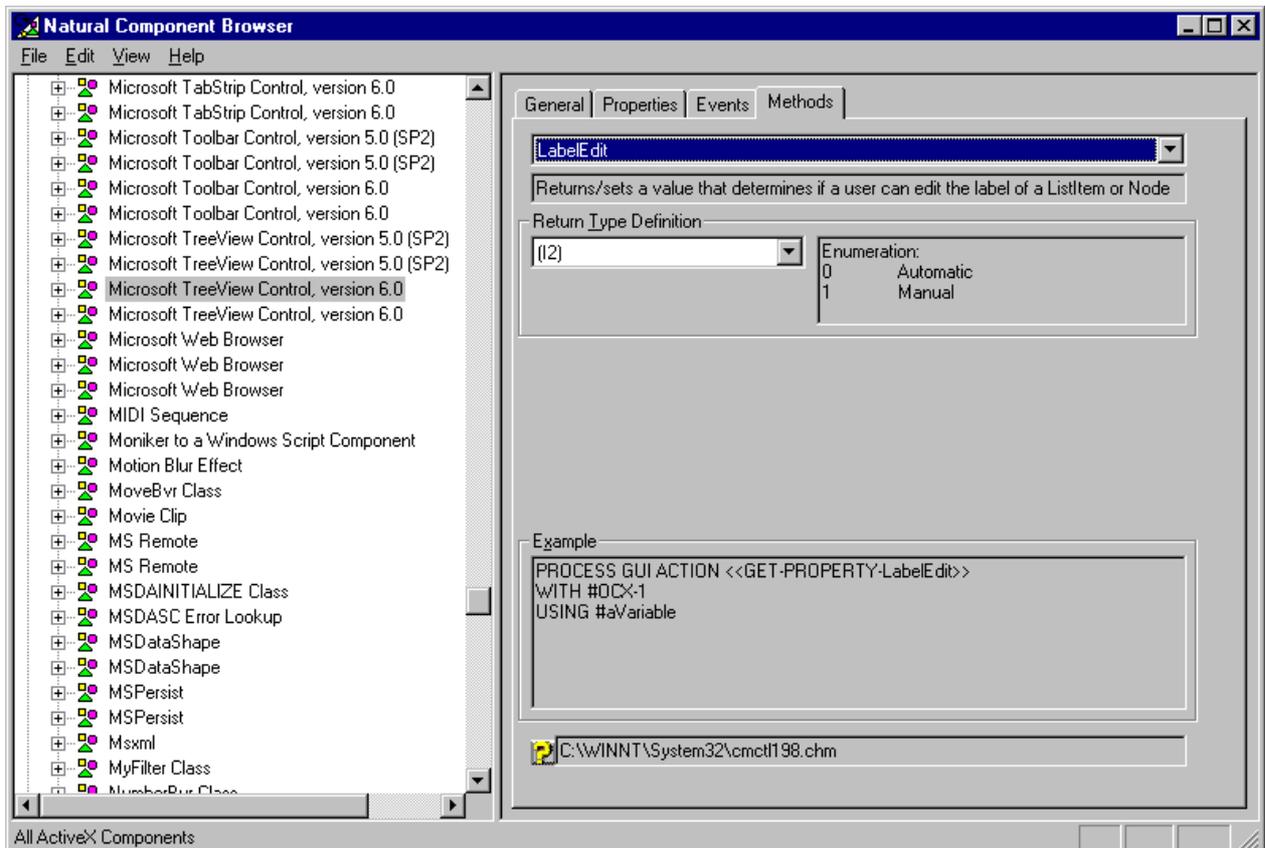
Events

The example on the page **Events** shows how to query event parameters by name or by position. Here #OCX-1 is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.



Methods

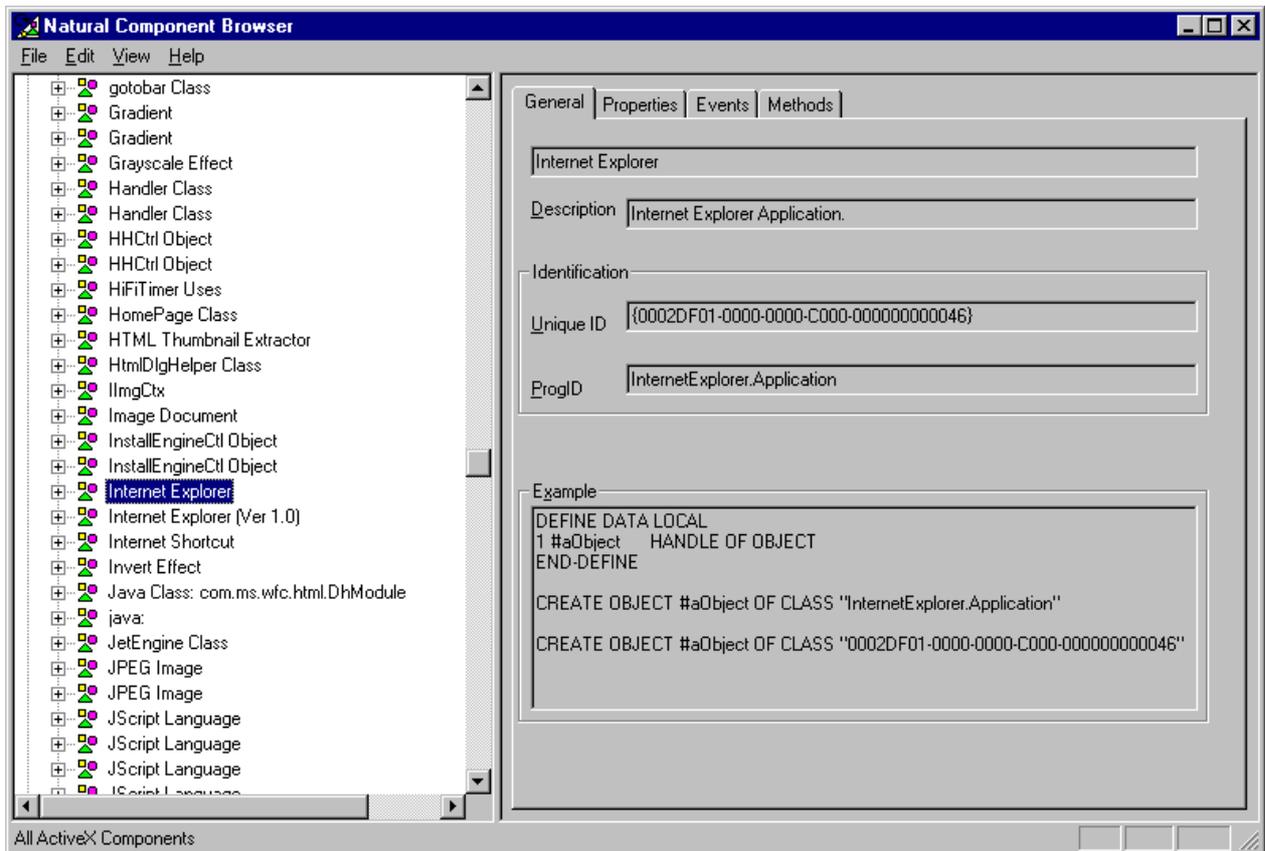
The example on the page **Methods** shows how to use methods, GET-properties and PUT-properties. Here #OCX-1 is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required. The actual parameter names are already inserted into the statement if they are available. Otherwise default parameter names P0, P1, ... are used as placeholders. These names can be replaced by application-specific variables.



Automation Objects

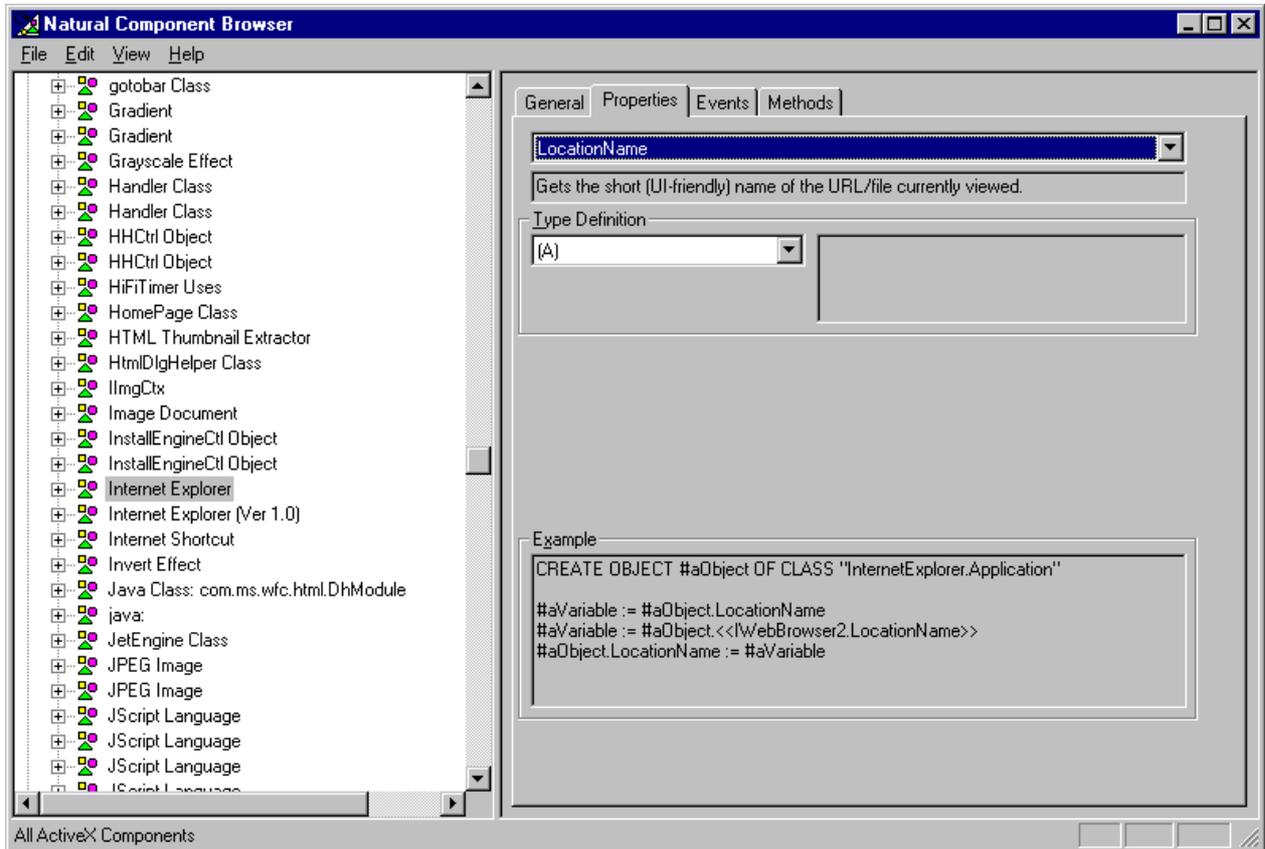
General

The example on the page **General** shows how an object of this type is instantiated. Here #aObject denotes a variable that can be adapted to the current application.



Properties

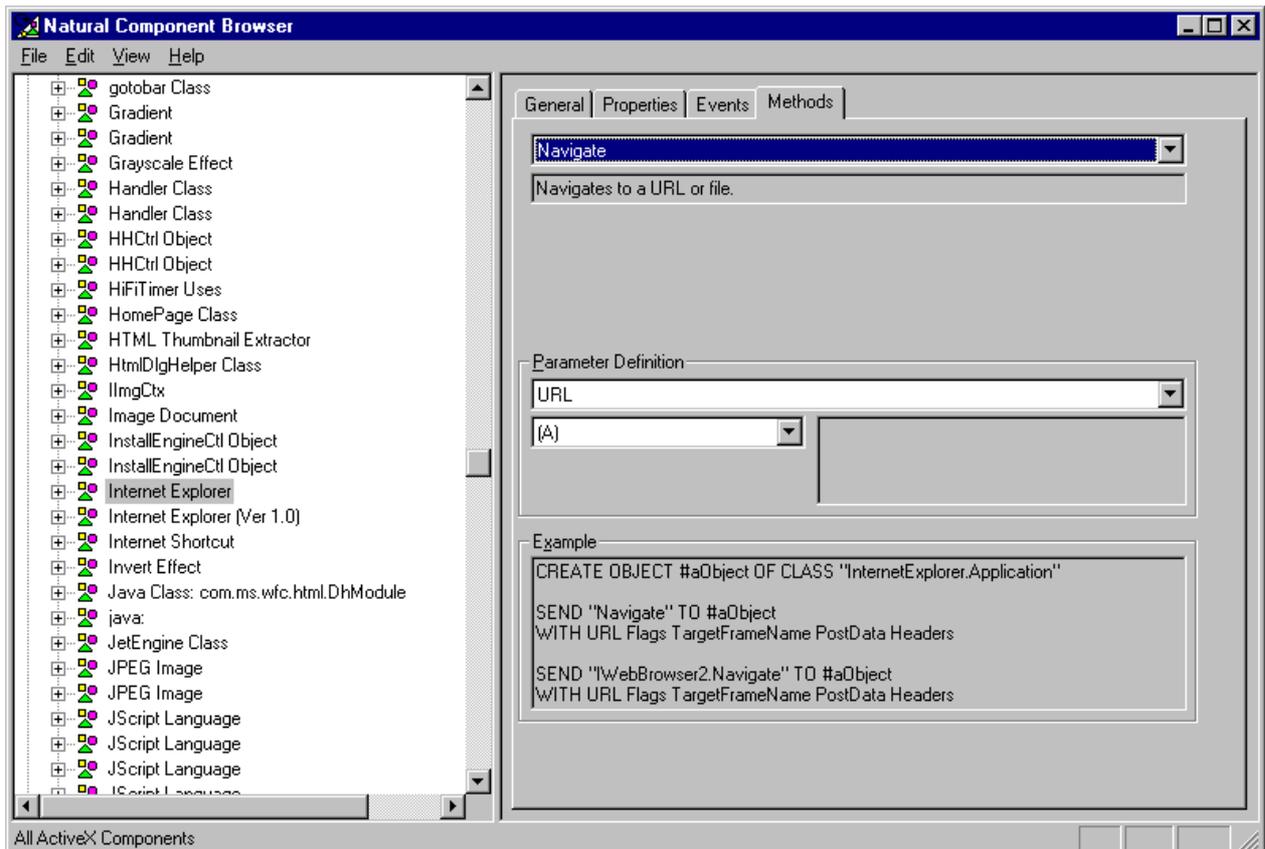
The example on the page **Properties** shows how to assign a property to a variable and how to assign a variable to a property. Here #aObject is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.



Methods

The example on the page **Methods** shows how to use methods. Here #aObject is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.

The actual parameter names are already inserted into the statement if they are available. Otherwise default parameter names P0, P1, ... are used as placeholders. These names can be replaced by application-specific variables.

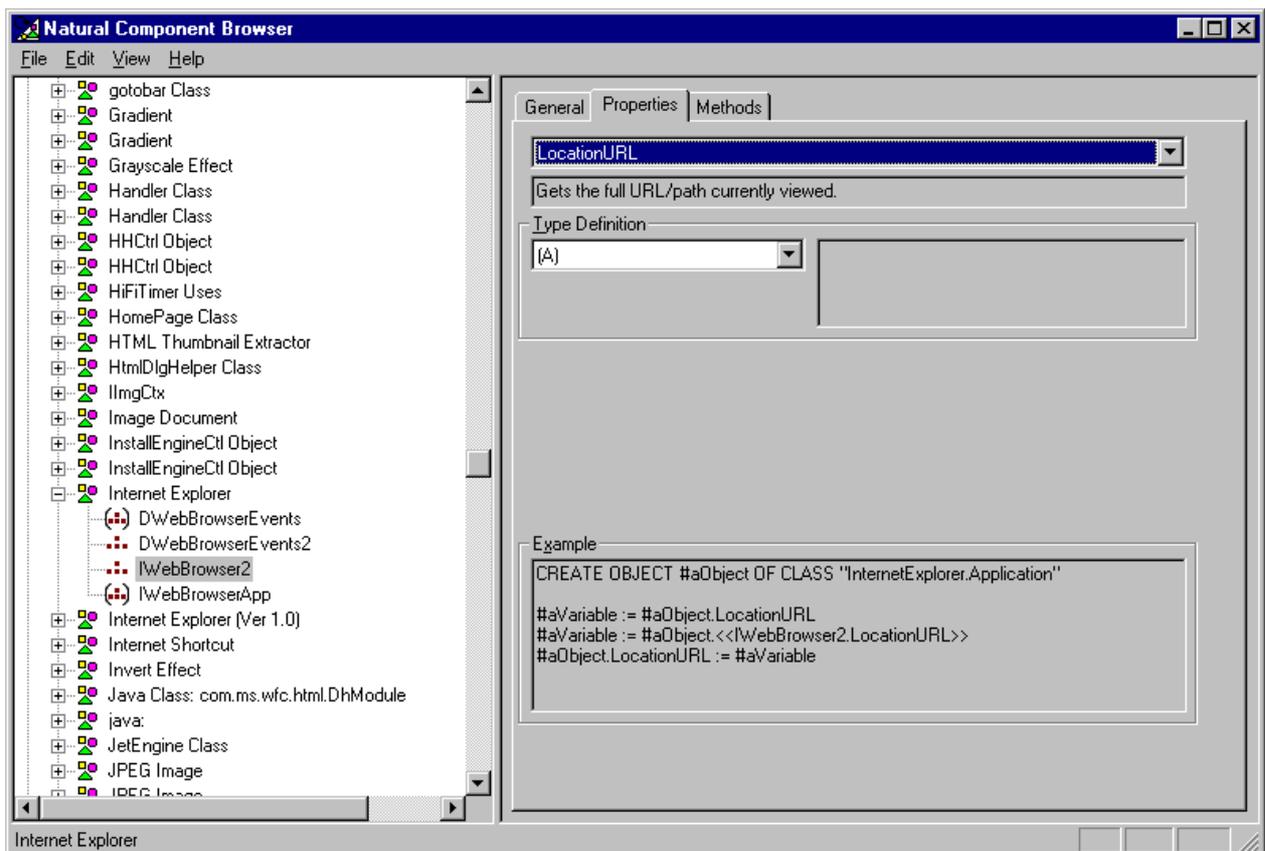


Interfaces

For interfaces that belong to group 'Interfaces' and that are not considered in the context of a class, the examples are generated as for Automation Objects. Only the CREATE OBJECT statement is left aside.

Properties

The example on the page **Properties** shows how to assign a property to a variable and how to assign a variable to a property. Here #aObject is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.



Methods

The example on the page **Methods** shows how to use methods. Here #aObject is the defined object handle and #aVariable refers to an application-specific variable. Both names can be adapted if required.

The actual parameter names are already inserted into the statement if they are available. Otherwise default parameter names P0, P1, ... are used as placeholders. These names can be replaced by application-specific variables.

