Astra® QuickTest™

Object Model Reference

Version 5.5

MERCURY INTERACTIVE

Astra QuickTest Object Model Reference, Version 5.5

If you have any comments or suggestions regarding this document, please send them via e-mail to documentation@merc-int.com.

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089 USA

AQTREF5.5/01

Table of Contents

Welcome to the Object Model Reference

Welcome to the QuickTest Object Model Reference.

This reference guide is intended to help you use QuickTest test objects, methods, and properties in your tests.

Using This Guide

QuickTest records your test in VBScript, which is Microsoft's Web scripting language. Your test script is a combination of standard VBScript statements and statements that use QuickTest *test objects*, methods and properties.

A test object is an object that QuickTest uses to represent an object in your application. An action performed on an object is a *method*. Each test object has one or more methods with which it is associated. Each test object also has a number of *properties* that describe the object.

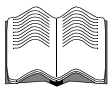
The QuickTest Object Model Reference includes:

- a description of each QuickTest test object
- a list of the methods and properties associated with each object
- a description of each method and property and the relevant syntax
- an example of usage for each method and property

For information on how to add statements using QuickTest test objects, methods and properties to your tests, refer to the "Enhancing Your Tests with Programming" chapter in the *Astra QuickTest User's Guide*.

To learn about working with VBScript, you can open Microsoft's VBScript Language Reference from the QuickTest Help menu. Note that you can edit and view your test in VBScript using Expert View. For additional information, refer to the "Testing in the Expert View" chapter in the *Astra QuickTest User's Guide*.

QuickTest Documentation Set



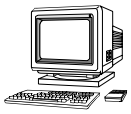
In addition to this reference guide, Astra QuickTest comes with a complete set of documentation:

Astra QuickTest Installation Guide explains how to install Astra QuickTest on a single computer or on a network.

Astra QuickTest Tutorial teaches you basic QuickTest skills and shows you how to start testing your applications.

Astra QuickTest User's Guide provides step-by-step instructions on how to use Astra QuickTest to test your Web site. It describes many useful testing tasks and options not covered in the tutorial.

Online Resources



Astra QuickTest includes the following online resources:

Read Me First (available from the Astra QuickTest Start menu program group) provides last-minute news and information about Astra QuickTest.

Astra QuickTest Tutorial (available from the Astra QuickTest Welcome window or the Astra QuickTest Help menu) teaches you basic QuickTest skills and shows you how to start testing your applications.

What's New in Astra QuickTest (available from the Astra QuickTest Help menu) describes the newest features and supported environments in the latest versions of Astra QuickTest.

Books Online (available from the Astra QuickTest Start menu program group or the Astra QuickTest Help menu) displays context-sensitive help for dialog boxes, the *Astra QuickTest Object Model Reference*, and the *Astra QuickTest User's Guide* in an HTML Help browser.

Mercury Tours sample Web site (available from the Astra QuickTest Browser Settings dialog box) is the basis for many examples in this book. The URL for this Web site is *http://mercurytours.mercuryinteractive.com*.

Astra QuickTest Context-Sensitive Help (available from dialog boxes and windows) describes QuickTest dialog boxes and windows.

Astra QuickTest Object Model Reference (available from the Astra QuickTest Help menu) an online version of this printed book.

VBScript Reference (available from the Astra QuickTest Help menu) describes Microsoft's VBScript language, and includes a tutorial and function reference.

Technical Support Online (available from the Astra QuickTest Help menu) uses your default Web browser to open Mercury Interactive's Customer Support Web site. This site enables you to browse the knowledge base and add your own articles, post to and search user discussion forums, submit support requests, download patches and updated documentation, and more. The URL for this Web site is *http://support.mercuryinteractive.com*.

Support Information (available from the Astra QuickTest Help menu) presents the locations of Mercury Interactive's Customer Support Web site and home page, the e-mail address for sending information requests, the name of the relevant news group, the location of Mercury Interactive's public FTP site, and a list of Mercury Interactive's offices around the world.

Mercury Interactive on the Web (available from the Astra QuickTest Help menu) uses your default Web browser to open Mercury Interactive's home page. This site provides you with the most up-to-date information on Mercury Interactive and its products. This includes new software releases, seminars and trade shows, customer support, educational services, and more. The URL for this Web site is *http://www.mercuryinteractive.com*.

Typographical Conventions

This book uses the following typographical conventions:

Bold	**Bold** text indicates method or function names.
Italics	*Italic* text indicates method or function arguments, file names or paths, and book titles.
<>	Angle brackets enclose a part of a file path or URL address that may vary from user to user (for example, *<MyProduct installation folder>\bin*.
Helvetica	The Helvetica font is used for examples and text that is to be typed literally.
[ ]	Square brackets enclose optional arguments.
...	In a line of syntax, an ellipsis indicates that more items of the same format may be included. In a programming example, an ellipsis is used to indicate lines of a program that were intentionally omitted.
|	A vertical bar indicates that one of the options separated by the bar should be selected.

1

ActiveX

The following objects can be used when testing ActiveX controls.

- ActiveX Object
- AcxTable Object
- AcxUtil Object

ActiveX Object

An ActiveX Control.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- GetROProperty Method
- MakeVisible Method
- MouseMove Method
- SetTOProperty Method
- Type Method

➤ WaitProperty Method

Associated Properties

➤ Object Property

Click Method

Clicks on an object.

Syntax

ActiveX.**Click** *x*, *y* [, *button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

ActiveX.**DblClick** *x*, *y* [, *button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

ActiveX.Drag *x*, *y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

ActiveX.Drop *x*, *y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	integer	**Optional.** The button is released in order to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

ActiveX.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on an ActiveX object.

Note: The event is sent to the container of the ActiveX object and does not affect the ActiveX object itself. For example, simulating a click event does not actually perform the click.

Syntax

ActiveX.FireEvent *EventName*, [[*EventArg(s)*],...]

Argument	Type	Description
EventName	string	Name of the event to simulate. The list of possible events depends on the ActiveX object.
EventArg(s)	N/A	Zero or more arguments of the event. The list of arguments depends on the *EventName*.

Example

The following example simulates a click event on the **Save** toolbar button.

```
Browser("Homepage").Page("Welcome").ActiveX("MyToolbar").FireEvent
"Click", "Save"
```

GetTOProperty Method

Returns the specified value of a property for a test object. The value is taken from the Object Repository.

Syntax

ActiveX.GetTOProperty(*Property*, [, *Flags*, *Type*, *ParameterName*]**)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

GetROProperty Method

Returns the current value of the specified run-time object property.

Syntax

ActiveX.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

MakeVisible Method

If the ActiveX control is not visible in the window, the **MakeVisible** method scrolls it into view.

Syntax

ActiveX.MakeVisible

Example

The following example brings the CIRC3 ActiveX control into view.

```
Browser("Welcome to A-Soft").Page("Untitled
Normal").ActiveX("CIRC3.Circ3Ctrl.1").MakeVisible
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

ActiveX.MouseMove *x*, *y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20,30) inside the "Advanced" button.

```
Browser("MyPage").Dialog("Settings").WinButton("Advanced").MouseMove 20,
30
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the test object properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

ActiveX.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty ("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in an ActiveX control.

Syntax

ActiveX.**Type** *string*

Argument	Type	Description
string	string	The text string.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

ActiveX.**WaitProperty(** *property*, *value*, *time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3.
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

ObjectName(*ObjectName*).Object .*Method_to_activate*( )
or

myObj=*ObjectName*(*ObjectName*).Object
myObj.*Method_to_activate*()

Example

In the following example, suppose the MakeObjVisible method is supported for your ActiveX control. To activate the MakeObjVisible method (method), you would insert the following statement into your test script:

```
Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Object.MakeObjVisible()
```

Alternatively you could insert the following:

```
MyObj=Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Object
MyObj.MakeObjVisible()
```

AcxTable Object

An ActiveX table.

Associated Methods:

- ActivateCell Method
- ActivateColumn Method
- ActivateRow Method
- Click Method
- DblClick Method

- Drag Method
- Drop Method
- Exist Method
- GetCellData Method
- GetTOProperty Method
- GetROProperty Method
- MakeVisible Method
- MouseMove Method
- SelectCell Method
- SetCellData Method
- SelectColumn Method
- SelectRow Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- ColumnCount Property
- Object Property
- RowCount Property

ActivateCell Method

Double-clicks the specified cell in a table.

Syntax

AcxTable.ActivateCell *Row*, *Col*

Argument	Type	Description
Row	string or number	The location of the row within the table.
Col	string or number	The location of the column within the table.

Example

The following example double clicks the third row and 1st column in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").ActivateCel
l "#3", "#1"
```

ActivateColumn Method

Double-clicks the specified column in a table.

Syntax

AcxTable.ActivateColumn *Col*

Argument	Type	Description
Col	string or number	The location of the column within the table.

Example

The following example double clicks the third column in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").ActivateCol
umn "#3"
```

ActivateRow Method

Double-clicks the specified row in a table.

Syntax

AcxTable.ActivateColumn *Row*

Argument	Type	Description
Row	string or number	The location of the row within the table.

Example

The following example double clicks the third row in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").ActivateRo
w "#3"
```

Click Method

Clicks on an object.

Syntax

AcxTable.**Click** *x*, *y* [, *button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47,131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

AcxTable.**DblClick** *x*, *y* [, *button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73,120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

AcxTable.Drag *x*, *y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

AcxTable.Drop *x*, *y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	integer	**Optional.** The button that is released in order to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

AcxTable.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetCellData Method

Retrieves the contents of the specified cell from a table.

Syntax

AcxTable.GetCellData(*Row***, *Col***)**

Argument	Type	Description
Row	string or number	The location of the row within the table.
Col	string or number	The location of the column within the table.

Example

The following example retrieves the contents from the third row and 1st column in the MSFlexGrid table, and displays them in a message box.

```
CellText =
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").GetCellDat
a("#3", "#1")
MsgBox CellText
```

GetTOProperty Method

Returns the specified value of a property for a test object. The value is taken from the Object Repository.

Syntax

AcxTable.GetTOProperty(*Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

GetROProperty Method

Returns the current value of the specified run-time object property.

Syntax

AcxTable.GetROProperty(*Property***, *in_PropData*)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

MakeVisible Method

If the AcxTable is not visible in the window, the **MakeVisible** method scrolls it into view.

Syntax

AcxTable.MakeVisible

Example

The following example brings the CIRC3 AcxTable into view.

```
Browser("Welcome to A-Soft").Page("Untitled
Normal").AcxTable("CIRC3.Circ3Ctrl.1").MakeVisible
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

AcxTable.MouseMove *x*, *y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" button.

```
Browser("MyPage").Dialog("Settings").WinButton("Advanced").MouseMove 20, 30
```

SelectCell Method

Selects (clicks) the specified cell in a table.

Syntax

AcxTable.SelectCell *Row*, *Col*

Argument	Type	Description
Row	string or number	The location of the row within the table.
Col	string or number	The location of the column within the table.

Example

The following example clicks the third row and first column in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").SelectCell
"#3", "#1"
```

SetCellData Method

sets the contents of a cell to the specified text in a table.

Syntax

AcxTable.SetCellData *Row*, *Col*, *Data*

Argument	Type	Description
Row	string or number	The location of the row within the table.
Col	string or number	The location of the column within the table.
Data	string	The contents to be entered into the specified cell.

Example

The following example enters the text hello into the third row and first column of the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").SetCellDat
a "#3", "#1", "Hello"
```

SelectColumn Method

Selects (clicks) the specified column in a table.

Syntax

AcxTable.SelectColumn *Col*

Argument	Type	Description
Col	string or number	The location of the column within the table.

Example

The following example clicks the first column in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").SelectColu
mn "#1"
```

SelectRow Method

Selects (clicks) the specified row in a table.

Syntax

AcxTable.SelectRow *Row*

Argument	Type	Description
Row	string or number	The location of the row within the table.

Example

The following example clicks the third row in the MSFlexGrid table.

```
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").SelectRow
"#3"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the test object properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

AcxTable.SetTOProperty *Property*, *Value*

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in an AcxTable.

Syntax

AcxTable.**Type** *string*

Argument	Type	Description
string	string	The text string.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

AcxTable.**WaitProperty(** *property*, *value*, *time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled",ON,3.
```

ColumnCount Property

Returns the number of columns in the table.

Syntax

AcxTable.ColumnCount

Example

The following example returns the number of columns in the MSFlexGrid table and displays the number in a message box.

```
Columns =
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").ColumnCo
unt
MsgBox Columns
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

ObjectName(*ObjectName*).Object.*Method_to_activate*()
or

myObj=*ObjectName*(*ObjectName*).Object
myObj.*Method_to_activate*()

Example

In the following example, suppose the MakeObjVisible method is supported for your ActiveX control. To activate the MakeObjVisible method (method), you would insert the following statement into your test script:

```
Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Object.MakeObjVisible()
```

Alternatively you could insert the following:

```
MyObj=Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Object
MyObj.MakeObjVisible()
```

RowCount Property

Returns the number of rows in the table.

Syntax

AcxTable.RowCount

Example

The following example returns the number of rows in the MSFlexGrid table and displays the number in a message box.

```
Rows =
Browser("Mercury").Page("FlexGridDemo").AcxTable("MSFlexGrid").RowCount
MsgBox Rows
```

AcxUtil Object

An ActiveX Utility object.

Associated Methods:

➤ FireEvent Method

Associated Properties

➤ Object Property

FireEvent Method

Simulates an event on an ActiveX object.

Note: The event is sent to the container of the ActiveX object and does not affect the ActiveX object itself. For example, simulating a click event does not actually perform the click.

Syntax

AcxUtil.FireEvent *Object*, *EventName*, [[*EventArg(s)*],...]

Argument	Type	Description
Object	string	The ActiveX object on which the event is simulated.
EventName	string	Name of the event to simulate. The list of possible events depends on the object.
EventArg(s)	N/A	Zero or more arguments of the event. The list of arguments depends on the event.

Example

The following example simulates a click event on the **Save** toolbar button.

```
Set ToolbarObj =
Browser("Homepage").Page("Welcome").ActiveX("MyToolbar").Object
AcxUtil.FireEvent ToolbarObj, "Click", "Save"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

EnvironmentObject(*ObjectName***).Object.***Method_to_activate***()**
or

myObj=EnvironmentObject(*ObjectName***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the MakeObjVisible method is supported for your ActiveX control. To activate the MakeObjVisible method (method), you would insert the following statement into your test script:

```
Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Object.Ma
keObjVisible()
```

Alternatively you could insert the following:

```
MyObj=Browser(BrowserName).Page(PageName).ActiveX(ActiveX_control).Ob
ject
MyObj.MakeObjVisible()
```

2

Flash

The following objects can be used when testing Macromedia Flash objects.

Note: Macromedia Flash objects are supported in QuickTest Professional only.

➤ FlashControl Object

FlashControl Object

A Macromedia Flash movie.

Associated Methods:

➤ ButtonAction Method

➤ ButtonClick Method

➤ ButtonOver Method

➤ ClipGotoFrame Method

➤ ClipGotoLabel Method

➤ ClipPlay Method

➤ ClipStop Method

➤ GetClipCurrentFrame Method

➤ GetClipCurrentLabel Method

➤ GetCurrentFrame Method

- GetVariable Method
- GotoFrame Method
- Play Method
- ProportionalClick Method
- ProportionalMove Method
- SetText Method
- SetVariable Method
- Stop Method
- WaitFrame Method

Associated Properties:

- Object Property

ButtonAction Method

Waits until the root clip on the specified level reaches the specified frame, and then reproduces the Flash Script actions associated with the specified event condition for the specified button.

Syntax

FlashControl.ButtonAction *ButtonName*, *Condition*, *Level*, *FrameNumber*

Argument	Type	Description
ButtonName	String	The logical name of the button if the button has a name. Otherwise the #ID. #ID is the tag ID that defines the button in the SWF file.
Condition	Number	The action condition flag. For a listing of condition flags, numeric values, and conditions, see Condition Flags.
Level	Number	The level number for the specified button.
FrameNumber	Number	The frame number in the specified level.

Example

The following example waits for frame 46 in the _level0 clip, and then performs the action with the code: "2"

```
Browser("Y E S").Page("Y E
S_2").Frame("top").FlashControl("ShockwaveFlash.Shock").ButtonAction
"#108", 2, 0, 46
```

Condition Flags

The following conditions can be used in the condition flag argument for the FlashControl ButtonAction method.

Button Action Condition	Numeric Value (decimal)	User Action
IdleToOverUp	1	Mouse cursor is placed in a FlashButton area without pressing the mouse button. (Roll over)
OverUpToIdle	2	Mouse cursor leaves a FlashButton area without pressing any mouse button. (Roll Out)
OverUpToOverDown	4	Mouse button is pressed while over a FlashButton. (Press)
OverDownToOverUp	8	Mouse button is released while over a FlashButton. (Release)
OverDownToOutDown	16	Mouse button is pressed while over a FlashButton and dragged out of the FlashButton area. (Drag Out)
OutDownToOverDown	32	Mouse button is pressed outside of a FlashButton area and then dragged into a FlashButton area
OutDownToIdle	64	N/A
IdleToOverDown	128	N/A
OverDownToIdle	256	Mouse button is released outside a FlashButton area.

ButtonClick Method

Waits while the root clip on given level reaches given frame and then reproduces Flash Script actions associated with the LBUTTONDOWN event for the specified button.

Syntax

FlashControl.ButtonClick *ButtonName*, *Level*, *FrameNumber*

Argument	Type	Description
ButtonName	String	The logical name of the button if the button has a name. Otherwise the #ID. #ID is the tag ID that defines the button in the SWF file.
Level	Number	The level number for the specified button.
FrameNumber	Number	The frame number in the specified level.

Example

The following example performs an LBUTTONDOWN event on the #11 button.

```
Browser("DOGBERT - Mercury").
Page("Calculator").FlashControl("ShockwaveFlash.Shock").ButtonClick "#11", 0,
0
```

Note: If the specified button resides in an internal clip, several **WaitFrame** methods will be recorded before **ButtonClick**.

ButtonOver Method

Waits until the root clip on the specified level reaches the specified frame, and reproduces Flash Script actions associated with the MOUSEOUTTOOVER event for the specified button.

Syntax

FlashControl.ButtonOver *ButtonName*, *Level*, *FrameNumber*

Argument	Type	Description
ButtonName	String	The logical name of the button if the button has a name. Otherwise the #ID. #ID is the tag ID that defines the button in the SWF file.
Level	Number	The level number for the specified button.
FrameNumber	Number	The frame number in the specified level.

Example

The following example performs an MOUSEOUTTOOVER event on the #11 button.

```
Browser("DOGBERT - Mercury").
Page("Calculator").FlashControl("ShockwaveFlash.Shock").ButtonOver "#11", 0,
0
```

Note: If the specified button resides in an internal clip, several WaitFrame methods will be recorded before **ButtonOver**.

ClipGotoFrame Method

Goes to the specified clip frame according to frame number.

Syntax

FlashControl.ClipGotoFrame *ClipName*, *FrameNum*

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.
FrameNum	Number	The frame number.

Example

The following example uses the **ClipGotoFrame** method to navigate to the 12th frame of the "myPlane" clip. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
Browser("moviecontainer").Page("moviecontainer").FlashControl("moviecontain
er").ClipGotoFrame "myContainer/myPlane", 12
```

ClipGotoLabel Method

Goes to the specified clip frame according to frame label.

Syntax

FlashControl.ClipGotoLabel *ClipName*, *FrameLabel*

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.
FrameLabel	String	The frame label.

Example

The following example uses the **ClipGotoLabel** method to navigate to the "Rotate" frame of the "myPlane" clip. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
Browser("moviecontainer").Page("moviecontainer").FlashControl("moviecontain
er").ClipGotoLabel "myContainer/myPlane", "Rotate"
```

ClipPlay Method

Begins play on a movie clip from the current location.

Syntax

FlashControl.ClipPlay *ClipName*

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.

Example

The following example uses the **ClipPlay** method to start play on the "myPlane" clip. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
Browser("moviecontainer").Page("moviecontainer").FlashControl("moviecontain
er").ClipPlay "myContainer/myPlane"
```

ClipStop Method

Stops play on a movie clip.

Syntax

FlashControl.ClipStop *ClipName*

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.

Example

The following example uses the **ClipStop** method to stop play on the "myPlane" clip. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
Browser("moviecontainer").Page("moviecontainer").FlashControl("moviecontain
er").ClipStop "myContainer/myPlane"
```

GetClipCurrentFrame Method

Retrieves the current frame number of a movie clip.

Syntax

FlashControl.GetClipCurrentFrame(*ClipName* **)**

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.

Return Value

Number

Example

The following example uses the **GetClipCurrentFrame** method to retrieve the current frame number in the "myPlane" clip. The frame number will be stored in the new variable *framenum*. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
framenum = Browser("moviecontainer").Page("moviecontainer").
FlashControl("moviecontainer").GetClipCurrentFrame("myContainer/myPlane")
```

GetClipCurrentLabel Method

Retrieves the label of the current frame.

Syntax

FlashControl.GetClipCurrentLabel(*ClipName* **)**

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.

Return Value

String

Example

The following example uses the **GetClipCurrentLabel** method to retrieve the current frame label in the "myPlane" clip. The frame label will be stored in the new variable, *framelabel*. The "myPlane" clip is a clip inside a clip called "myContainer", both of which reside inside the movie called "moviecontainer".

```
framelabel = Browser("moviecontainer").Page("moviecontainer").
FlashControl("moviecontainer").GetClipCurrentLabel("myContainer/myPlane")
```

GetCurrentFrame Method

Retrieves the current frame number of a movie clip.

Syntax

FlashControl.GetCurrentFrame(*ClipName* **)**

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.

Return Value

Number

Example

The following example uses the **GetCurrentFrame** method to retrieve the current frame in the main (_level0) clip. The frame label will be stored in the new variable, CurrentRootFrame.

```
CurrentRootFrame =
Browser("FrameAction").Page("FrameAction").FlashControl("ShockwaveFlash.
Shock").GetCurrentFrame
```

GetVariable Method

Retrieves the value of the specified movie variable.

Syntax

FlashControl.GetVariable *Variable*

Argument	Type	Description
Variable	String	The full name of the variable. Use the standard Macromedia syntax for entering the variable name.

Return Value

String

Example

The following example uses the **GetVariable** method to retrieve the value for the "another" variable, and stores the value in the new variable, *varvalue*.

```
varvalue = Browser("textfield").Page("textfield").
FlashControl("textfield").GetVariable("myContainer/another")
```

GotoFrame Method

Goes to the specified frame of the root clip according to frame number.

Syntax

FlashControl.GotoFrame *FrameNum*

Argument	Type	Description
ClipName	String	The full name of the clip.
FrameNum	Number	The frame number

Example

The following example uses the **GotoFrame** method to navigate to the 5th frame in the main (_level0) movie clip.

```
Browser("Tektronix Color").Page("Tektronix
Color").Frame("BODY").FlashControl("printer").GotoFrame 5
```

Play Method

Begins play on a movie from the current location.

Syntax

FlashControl.Play

Example

The following example uses the **Play** method to begin playing the movie.

```
Browser("Tektronix Color").Page("Tektronix
Color").Frame("BODY").FlashControl("printer").Play
```

ProportionalClick Method

Clicks on the FlashControl object based on the relative location of the click proportional to the size of the entire screen.

Note: This method is applicable only on interactive video clips.

Syntax

FlashControl.ProportionalClick(*PercentX*, *PercentY*, [*button*] **)**

Argument	Type	Description
Percent X, Percent Y	number	The relative coordinates of the click as a percentage of the width (X) and height(Y) of the screen.
button	LEFT or RIGHT	**Optional.** The mouse button performing the click. Default = LEFT

Example

The following example uses the **ProportionalClick** method to click on the clip in a location 2.20% into the screen, and 12.1% down from the top.

```
FlashControl("Top Stories").ProportionalClick 20.2, 12.1
```

ProportionalMove Method

Waits the specified amount of time and then moves the mouse within the Flash object based on the relative location of the mouse proportional to the size of the entire screen.

Note: This method is applicable only on interactive video clips.

Syntax

FlashControl.ProportionalMove(*PercentX*, *PercentY*, [*wait_time*] **)**

Argument	Type	Description
Percent X, Percent Y	number	The relative coordinates of the click as a percentage of the width (X) and height(Y) of the screen.
wait_time	LEFT or RIGHT	**Optional.** The amount of time to wait before performing the move. The amount of time is measured from the shorter of: ➤ start of play or start of record (if this is the first ProportionalMove in the test) ➤ The previous ProportionalMove event

Example

The following example uses the **ProportionalMove** method to move the mouse to the location 22.1% into the screen, and 12.21% down from the top. The move occurs 3.2 seconds after the previous ProportionalMove.

```
FlashControl("Top Stories").ProportionalMove 22.1, 12.21, 3.2
```

SetText Method

Sets the specified text in the appropriate Flash text edit control.

Syntax

FlashControl.SetText *Name*, *Text*

Argument	Type	Description
Name	String	The full name of the variable for the text control. Use the standard Macromedia syntax.
Text	String	The text to be set.

Example

The following example inserts the text: "5" in the "_level0:display" text edit control.

```
Browser("DOGBERT - Mercury").Page("Calculator").
FlashControl("ShockwaveFlash.Shock").SetText "_level0:display", "5"
```

SetVariable Method

Sets the value of the specified movie variable.

Syntax

FlashControl.SetVariable *Variable*, *Value*

Argument	Type	Description
Variable	String	The name of the variable to be set.
Value	String	The value to be assigned to the variable.

Example

The following example uses the **SetVariable** method to set the value for the "myfield" variable to "NewValue".

```
Browser("textfield").Page("textfield").FlashControl("textfield").SetVariable
"myField", "NewValue"
```

Stop Method

Stops play and returns to the beginning of the movie.

Syntax

FlashControl.Stop

Example

The following example uses the **Stop** method to stop the "printer" movie.

```
Browser("Tektronix Color").Page("Tektronix
Color").Frame("BODY").FlashControl("printer").Stop
```

WaitFrame Method

Waits until the movie clip reaches the specified frame in the specified clip before performing the next step. This method is added automatically during recording.

Note: The **WaitFrame** step is not displayed in the Test Results tree.

Syntax

FlashControl.WaitFrame *ClipName*, *FrameNumber*

Argument	Type	Description
ClipName	String	The full name of the clip. Use the standard Macromedia syntax for entering the clip name.
FrameNumber	Number	The frame number in the specified clip.

Example

The following example uses the **WaitFrame** method to suspend all further action until the clip reaches the specified frame.

```
Browser ("DOGBERT - Mercury").Page
("Calculator").FlashControl("ShockwaveFlash.Shock").WaitFrame
"_level5/MyClip", 28
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Syntax

EnvironmentObject(ObjectName).Object.Method_to_activate()
or

myObj=EnvironmentObject(ObjectName).Object
myObj.Method_to_activate()

Example

In the following example, suppose the Rewind method is supported for your Flash control. To activate the Rewind method (method), you would insert the following statement into your test script:

```
FlashControl("asd").Object.Rewind()
```
Alternatively you could insert the following:

```
myObj = ...FlashControl("asd").Object()
myObj.Rewind
```

3

Java

The following objects can be used when testing Java objects.

Note: Java objects are supported in QuickTest Professional only.

- Applet Object
- JavaButton Object
- JavaCheckBox Object
- JavaEdit Object
- JavaInternalFrame Object
- JavaList Object
- JavaMenu Object
- JavaObject Object
- JavaRadioButton Object
- JavaSlider Object
- JavaSpin Object
- JavaTab Object
- JavaTable Object
- JavaToolBar Object

Applet Object

A Java Applet.

Associated Methods:

- Activate Method
- Click Method
- Close Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- Maximize Method
- Minimize Method
- MouseDrag Method
- Move Method
- GetROProperty Method
- Resize Method
- Restore Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Activate Method

Activates the applet, internal frame, edit box or an item in a Java List.

Syntax

Applet.Activate

Example

The following example uses the **Activate** method to activate the puzzle.html applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Activate
```

Click Method

Clicks the specified location with the specified mouse button.

Syntax

Applet.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the periodic.html applet.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").Click 503, 55,
"LEFT"
```

Close Method

Closes the applet or internal frame.

Syntax

Applet.Close

Example

The following example uses the **Close** method to close the puzzle.html applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Close
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

Applet.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **)**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	required arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect = Browser("Periodic").Page("Periodic").Applet("Periodic").CreateObject
("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

Applet.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the periodic.html applet.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

Applet.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the login applet. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

Applet.FireEvent *constant* [**,** *EventParam(s)*]

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the event constructor except for the source and the EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "mybuttonapplet" applet.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").FireEvent
micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

Applet.FireEventEx *JavaClassName*, *EventID* [, *constructor_param3*,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "mybuttonapplet" applet.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").FireEventEx
"java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4,
4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Applet.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the name of the applet.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Maximize Method

Maximizes a resizable applet or internal frame.

Syntax

Applet.Maximize

Example

The following example uses the **Maximize** method to maximize the puzzle.html applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Maximize
```

Minimize Method

Minimizes a resizable applet or internal frame.

Syntax

Applet.Minimize

Example

The following example uses the **Minimize** method to minimize the "puzzle.html" applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Minimize
```

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

Applet.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").MouseDrag 4, 10, 10, 10, "LEFT"
```

Move Method

Moves an applet or an internal frame in the screen area.

Syntax

Applet.Move *x*, *y*

Argument	Type	Description
x, y	variant	Indicates the coordinates of the location to which the Applet should be moved.

Example

The following example uses the **Move** method to move the puzzle.html applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Move 10, 10
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Applet.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the applet.

```
CheckState = Browser.Page("Table Sample").Applet("puzzle.html").
GetROProperty ("Value") ' CheckState contains "OFF"
```

Resize Method

Resizes a resizable applet or internal frame to the specified dimensions.

Syntax

Applet.Resize *width*, *height*

Argument	Type	Description
width, height	variant	Indicates the new size of the applet.

Example

The following example uses the **Resize** method to resize the puzzle.html applet.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Resize 350,
500
```

Restore Method

Restores the applet or internal frame to its previous size.

Syntax

Applet.Restore

Example

The following example uses the **Restore** method to restore the puzzle.html applet to its previous size.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").Restore
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Applet.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of the applet.

```
Browser("Mercury Tours").Page("FindFlights").Applet("puzzle.html").
SetTOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

Applet.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the applet.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

```
Set myObj=JavaObjectName(ObjectName).Object
Set myObj.Method_to_activate()
```

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaButton Object

A Java Button.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaButton.Click

Example

The following example uses the **Click** method to click on the "One" button in the SwingSet demo_3 applet.

```
Browser("SwingSet demo").Page("SwingSet
demo_3").Applet("SwingSetApplet").JavaButton("One").Click
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaButton.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaButton("OK").
CreateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaButton.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaButton object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaObject("Period
ic").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the the "Login" Java button. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaButton("Login").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaButton.FireEvent *constant* [**,** *EventParam(s)*]

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyButton_0" JavaButton.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaButton("MyButton_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaButton.FireEventEx *JavaClassName*, *EventID* [, $constructor\_param_3$,..., $contructor\_param_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyButton_0" JavaButton.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaButton"My
Button_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0,
"BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaButton.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the name of the JavaButton.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaButton("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1, y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaButton.MouseDrag *x 1*, *y 1*, *x 2*, *y 2*, [*button*]

Argument	Type	Description
x_1, y_1	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, y_2	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaButton("PushMe"
).MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaButton.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaButton.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaButton("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaButton.SetTOProperty *Property*, *Value*, [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaButton.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaButton("puzzle.html").Set
TOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object..

Syntax

JavaButton.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaButton.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaButton("OK").Type
"ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(ObjectName).Object.Method_to_activate()
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set
MyObj=Browser("Browser").Page("PushButtonDemo").Applet("pushbutton.html"
).JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaCheckBox Object

A Java CheckBox.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaCheckBox.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaCheckBox.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaCheckBox("Periodic").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaCheckBox.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$..$consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaCheckBox("Status"
).CreateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaCheckBox.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaCheckBox.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaCheckBox("Pe
riodic").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaCheckBox.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "Yes" Java check box. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaCheckbox("Yes").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaCheckBox.FireEvent *constant* [**,** *EventParam(s)*]

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyCheckBox_0" JavaCheckBox.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaCheckBox
("MyCheckBox_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1,
"OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaCheckBox.FireEventEx *JavaClassName*, *EventID*
[, *constructor_param3*,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyCheckBox_0" JavaCheckBox.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaCheckBox
("MyCheckBox_0").FireEventEx "java.awt.event.MouseEvent",
"MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaCheckBox.GetTOProperty(*Property*[**,** *Flags***,** *Type***,** *ParameterName*]**)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaCheckBox name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaCheckBox("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaCheckBox.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaCheckBox("Periodic").MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaCheckBox.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaCheckBox.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaCheckBox("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

Set Method

Sets the value or state of the JavaCheckBox.

Syntax

JavaCheckBox.Set *SetValue*

Argument	Type	Description
SetValue	String	The value or state to be assigned to the object.

Example

The example below uses the **Set** method to set the state of the Enabled check box.

```
Browser("SwingSet demo").Page("SwingSet
demo_9").Applet("SwingSetApplet").JavaCheckBox("Enabled").Set "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaCheckBox.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaCheckBox.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaCheckBox("puzzle.html").
SetTOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaCheckBox.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaCheckBox.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaCheckBox("Status"
).Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaEdit Object

A Java Edit box.

Associated Methods:

- Activate Method
- Click Method
- CreateObject Method
- DblClick Method
- Delete Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- Insert Method
- MouseDrag Method
- GetROProperty Method

- Set Method
- SetInsertPos Method
- SetSelection Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Activate Method

Activates the applet, internal frame, edit box or an item in a Java List.

Syntax

JavaEdit.Activate

Example

The following example uses the **Activate** method to activate the Puzzle text box.

```
Browser("Crossword").Page("Crossword").Applet("puzzle.html").JavaEdit("Puzzl
e").Activate
```

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaEdit.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaEdit box.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaEdit("Periodic"
).Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaEdit.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$.. $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaEdit("Text").Create
Object ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaEdit.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaEdit object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaEdit("Periodic"
).DblClick "503", "55", "LEFT"
```

Delete Method

Deletes the specified text.

Syntax

JavaEdit.Delete *row*$_1$, *col*$_1$, *row*$_2$, *col*$_2$

Argument	Type	Description
row$_1$, *col*$_1$	variant	The row and column (top left) locations of the first character you want to delete.
row$_2$, *col*$_2$	variant	The row and column (bottom right) locations of the last character you want to delete.

Example

The following example uses the **Delete** method to delete the text, ",if you want," from the "Text:" edit field in the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Text:").Delet
e 1, 20, 1, 32
```

Exist Method

Checks that an object exists.

Syntax

JavaEdit.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaEdit.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyEditBox_0" JavaEdit.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaEdit("MyE
ditBox_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaEdit.FireEventEx *JavaClassName*, *EventID* [, *constructor_param3*,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyEditBox_0" JavaEdit.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaEdit("MyEditBox_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaEdit.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaEdit name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaEdit("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Insert Method

Inserts a text string into the specified location in a JavaEdit object.

Syntax

JavaEdit.Insert *string*, *row*, *column*

Argument	Type	Description
string	variant	The text string that you want to insert.
row, column	variant	The row and column that identify the exact location where you want to insert the text string. For single line edit fields, enter 0 for row value.

Example

The following example uses the **Insert** method to insert the text, ",if you want," into the "Text:" Editbox in the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Text:").Inser
t ",if you want,", 1, 32
```

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaEdit.MouseDrag *x_1*, *y_1*, *x_2*, *y_2*, [*button*]

Argument	Type	Description
x_1, *y_1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, *y_2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the Periodic JavaEdit object.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Periodic").M
ouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaEdit.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaEdit.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaEdit("puzzle.html").GetROProperty ("Value")
' CheckState contains "OFF"
```

Set Method

Sets the value or state of the JavaEdit object.

Syntax

JavaEdit.Set *SetValue*

Argument	Type	Description
SetValue	String	The value or state to be assigned to the object.

Example

The following example uses the **Set** method to set the text of the Your Message Java edit box.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Your
Message:").Set "Hi world."
```

SetInsertPos Method

Places the cursor in the specified location in a JavaEdit object.

Syntax

JavaEdit.SetInsertPos *row*, *column*

Argument	Type	Description
row, column	variant	The row and column that identify the exact location where you want to place the cursor within the edit box. For single line edit fields, enter 0 for row value.

Example

The following example uses the **SetInsertPos** method to place the cursor at the second character of the first row of the "Text:" Editbox in the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Text:").SetIn
sertPos 0, 2
```

SetSelection Method

Selects the specified text in the JavaEdit object. This is recorded when dragging the mouse and highlighting a text string, or by double-clicking a single word in the edit box.

Syntax

JavaEdit.SetSelection *row*$_1$, *col*$_1$, *row*$_2$, *col*$_2$

Argument	Type	Description
row$_1$, *col*$_1$	variant	The row and column (top left) locations of the first character you want to select.
row$_2$, *col*$_2$	variant	The row and column (bottom right) locations of the last character you want to select.

Example

The following example uses the **SetSelection** method to select the text, ",if you want," from the "Text:" edit field in the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaEdit("Text:").SetS
election 1, 20, 1, 32
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaEdit.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaEdit object.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaEdit("puzzle.html").SetTO
Property "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaEdit.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaEdit.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaEdit("Text").Type
"ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

```
Set myObj=JavaObjectName(ObjectName).Object
Set myObj.Method_to_activate()
```

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaInternalFrame Object

An internal frame that can be activated from the Java Applet.

Associated Methods:

- Activate Method
- Click Method
- Close Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetROProperty Method
- GetTOProperty Method
- Maximize Method
- Minimize Method
- MouseDrag Method
- Move Method
- Resize Method
- Restore Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Activate Method

Activates the applet, internal frame, edit box or an item in a Java List.

Syntax

JavaInternalFrame.Activate

Example

The following example uses the **Activate** method to activate the internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Activate
```

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaInternalFrame.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaInternalFrame object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaInternalFrame(
"Internal Frame").Click 503, 55, "LEFT"
```

Close Method

Closes the applet or internal frame.

Syntax

JavaInternalFrame.Close

Example

The following example uses the **Close** method to close the internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Close
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaInternalFrame.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaInternalFrame("Inte
rnal Frame").CreateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaInternalFrame.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the "Internal Frame" JavaInternalFrame object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaInternalFrame(
"Internal Frame").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaInternalFrame.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java internal frame. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").
JavaInternalFrame("username").Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaInternalFrame.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "Internal Frame" JavaInternalFrame.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaInternalFr
ame("Internal Frame").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1,
"OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaInternalFrame.FireEventEx *JavaClassName*, *EventID*
[, *constructor_param*$_3$,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "Internal Frame" JavaInternalFrame.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaInternalFr
ame("Internal Frame").FireEventEx "java.awt.event.MouseEvent",
"MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaInternalFrame.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaInternalFrame.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaInternalFrame("puzzle.html").GetROPropert
y ("Value") ' CheckState contains "OFF"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaInternalFrame.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaInternalFrame name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaInternalFrame("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Maximize Method

Maximizes a resizable applet or internal frame.

Syntax

JavaInternalFrame.Maximize

Example

The following example uses the **Maximize** method to maximize the "Internal Frame" internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Maximize
```

Minimize Method

Minimizes a resizable applet or internal frame.

Syntax

JavaInternalFrame.Minimize

Example

The following example uses the **Minimize** method to minimize the "Internal Frame" internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Minimize
```

MouseDrag Method

Performs a mouse drag operation from the specified x1, y1 coordinates to the specified x2, y2 coordinates.

Syntax

JavaInternalFrame.MouseDrag x_1, y_1, x_2, y_2, [*button*]

Argument	Type	Description
x_1, y_1	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, y_2	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaInternalFrame("In
ternal Frame").MouseDrag 4, 10, 10, 10, "LEFT"
```

Move Method

Moves an applet or internal frame in the screen area.

Syntax

JavaInternalFrame.Move *x*, *y*

Argument	Type	Description
x, y	variant	Indicates the coordinates of the location to which the Applet should be moved.

Example

The following example uses the **Move** method to move the internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Move 45, 52
```

Resize Method

Resizes a resizable applet or internal frame to the specified dimensions.

Syntax

JavaInternalFrame.Resize *width*, *height*

Argument	Type	Description
width, height	variant	Indicates the new size of the Internal frame.

Example

The following example uses the **Resize** method to resize the "Internal Frame" internal frame.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Resize 200, 200
```

Restore Method

Restores the applet or internal frame to its previous size.

Syntax

JavaInternalFrame.Restore

Example

The following example uses the **Restore** method to restore the "Internal Frame" internal frame to its previous size.

```
Browser("SwingSet demo").Page("SwingSet
demo_16").Applet("SwingSetApplet").JavaInternalFrame("Internal
Frame").Restore
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaInternalFrame.SetTOProperty *Property*, *Value,* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaInternalFrame.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaInternalFrame("puzzle.ht
ml").SetTOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaInternalFrame.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaInternalFrame.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaInternalFrame("Inte
rnal Frame").Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object
Set rect=MyObj.getBounds()
```

JavaList Object

A Java List box with single or multiple selection.

Associated Methods:

- Activate Method
- AddRange Method
- Click Method
- Collapse Method
- CreateObject Method
- DblClick Method
- Deselect Method
- DeselectRange Method
- Exist Method
- Expand Method
- ExtendSelect Method
- FireEvent Method

- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- Select Method
- SelectRange Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Activate Method

Activates an item in a Java List.

Syntax

JavaList.Activate *Item*

Argument	Type	Description
Item	String	Indicates the item to activate within the JavaList. This parameter is not required when using Activate with the Applet, JavaInternalFrame or JavaEdit objects.

Example

The following example uses the **Activate** method to activate the hotdog item of the ListPanel$1 list.

```
Browser("SwingSet demo").Page("SwingSet
demo_7").Applet("SwingSetApplet").JavaList("ListPanel$1").Activate "Hotdog"
```

AddRange Method

Adds the indicated range of items to the current selection in a list.

Syntax

JavaList.AddRange *Item*$_1$, *Item*$_2$

Argument	Type	Description
Item$_1$	variant	The first item in the range of items to be added to the list.
Item$_2$	variant	The last item in the range of items to be added to the list.

Example

The following example uses the **AddRange** method to add the range of items to those already selected.

```
Browser("SwingSet demo").Page("SwingSet
demo_4").Applet("SwingSetApplet").JavaList("ListPanel$1").AddRange "Ice
Cream", "Cherry Pie"
```

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaList.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaList object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaList("Periodic")
.Click 503, 55, "LEFT"
```

Collapse Method

Collapses an expandable Java List.

Syntax

JavaList.Collapse *Item*

Argument	Type	Description
Item	variant	Indicates the item to collapse within the JavaList.

Example

The following example uses the **Collapse** method to collapse the "Jazz" subfolder under the "Music" folder in the SwingSetApplet applet.

```
Browser("SwingSet demo").Page("SwingSet
demo_8").Applet("SwingSetApplet").JavaList("JTree").Collapse "Music;Jazz"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaList.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaList("Panel").Creat
eObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaList.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaList object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaList("Periodic")
.DblClick 503, 55, "LEFT"
```

Deselect Method

Deselects an item from a Java List.

Syntax

JavaList.Deselect *Item*

Argument	Type	Description
Item	variant	Indicates the item to deselect from the JavaList.

Example

The following example uses the **Deselect** method to deselect the "Hamburger" item from the "Entres" list within the "foodlist" applet.

```
Browser("FastFoodHaven").Page("KidsMeals").Applet("foodlist.html").JavaList("
Entres").Deselect "Hamburger"
```

DeselectRange Method

Deselects the indicated range of items from a list.

Syntax

JavaList.DeselectRange *item*$_1$**,** *item*$_2$

Argument	Type	Description
item$_1$	variant	The name of the first item in the range to be deselected.
item$_2$	variant	The name of the last item in the range to be deselected.

Example

The following example uses the **DeselectRange** method to deselect a range of objects from the "JTree" list.

```
Browser("SwingSet demo").Page("SwingSet
demo_4").Applet("SwingSetApplet").JavaList("ListPanel$1").DeselectRange
"Fries", "Cola"
```

Exist Method

Checks that an object exists.

Syntax

JavaList.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "profile" Java list box. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaList("profile").
Exist Then
msgbox("The object exists.")
```

Expand Method

Expands an expandable Java List.

Syntax

JavaList.Expand *Item*

Argument	Type	Description
Item	variant	Indicates the item to expand within the JavaList.

Example

The following example uses the **Expand** method to expand the "Jazz" subfolder under the "Music" folder.

```
Browser("SwingSet demo").Page("SwingSet
demo_8").Applet("SwingSetApplet").JavaList("JTree").Expand "Music;Jazz"
```

ExtendSelect Method

Selects an additional item from the list, such as when a user holds the control button in order to select multiple, non-consecutive items from a list.

Syntax

JavaList.ExtendSelect *item*

Argument	Type	Description
item	variant	The name of the item to be selected from the list.

Example

The following example uses the **ExtendSelect** method to select multiple, non-consecutive items on a list.

```
Browser("FastFoodHaven").Page("KidsMeals").Applet("foodlist.html").JavaList("
Entres").Select "Hamburger"
Browser("FastFoodHaven").Page("KidsMeals").Applet("foodlist.html").JavaList("
Entres").ExtendSelect "ChickenSandwhich"
Browser("FastFoodHaven").Page("KidsMeals").Applet("foodlist.html").JavaList("
Entres").ExtendSelect "HotDog"
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaList.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyList_0" JavaList.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaList("MyList_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaList.FireEventEx *JavaClassName*, *EventID* [, *constructor_param3*,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyList_0" JavaList.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaList("MyList_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaList.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaList name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaList("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x1,y1 coordinates to the specified x2, y2 coordinates.

Syntax

JavaList.MouseDrag x_1, y_1, x_2, y_2, [*button*]

Argument	Type	Description
x_1, y_1	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, y_2	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaList("Periodic").MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaList.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaList.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaList("puzzle.html").GetROProperty ("Value")
' CheckState contains "OFF"
```

Select Method

Selects an item from a Java List.

Syntax

JavaList.Select *Item*

Argument	Type	Description
Item	variant	Indicates the item to select within the JavaList.

Example

The following example uses the **Select** method to select the "Hamburger" item from the "Entres" list within the "foodlist" applet.

```
Browser("FastFoodHaven").Page("KidsMeals").Applet("foodlist.html").JavaList("
Entres").Select "Hamburger"
```

SelectRange Method

Selects the indicated range of items from a list.

Syntax

JavaList.SelectRange *item*$_1$, *item*$_2$

Argument	Type	Description
item$_1$	variant	The name of the first item in the range to be selected.
item$_2$	variant	The name of the last item in the range to be selected.

Example

The following example uses the **SelectRange** method to select a range of items from the list.

```
Browser("SwingSet demo").Page("SwingSet
demo_4").Applet("SwingSetApplet").JavaList("ListPanel$1").SelectRange
"Fries", "Hotdog"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaList.SetTOProperty *Property*, *Value*, [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaList.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaList("puzzle.html").SetTO
Property "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaList.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaList.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaList("Panel").Type
"ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

Set myObj=*JavaObjectName*(*ObjectName*).Object
Set myObj.*Method_to_activate*()

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaMenu Object

A Java Menu.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Methodg
- GetROProperty Method
- Select Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaMenu.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaMenu object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaMenu("Periodi
c").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaMenu.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaMenu("Panel").Cre
ateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaMenu.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaMenu object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaMenu("Periodi
c").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaMenu.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java menu. If the object exists a message box appears confirming its existence.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaMenu("username").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaMenu.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyMenu_0" JavaMenu.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaMenu("MyMenu_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaMenu.FireEventEx *JavaClassName*, *EventID* [, *constructor_param3*,..., *contructor_paramX*]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyMenu_0" JavaMenu.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaMenu("My
Menu_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0,
"BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaMenu.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaMenu name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaMenu("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1, y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaMenu.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaMenu("Periodic").
MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaMenu.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaMenu.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaMenu("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

Select Method

Selects a Java Menu item.

Syntax

JavaMenu.Select

Example

The following example selects the "First item" item within the "Text menu" menu:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaMenu("Text menu").JavaMenu("First
item").Select
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaMenu.SetTOProperty *Property*, *Value*, [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaMenu.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaMenu("puzzle.html").SetT
OProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaMenu.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaMenu.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaMenu("Panel").Typ
e "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaObject Object

A generic Java object.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- SetTOProperty Method
- Type Method

Associated Properties:

➤ Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaObject.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaObject object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaObject("Period
ic").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaObject.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaObject("Panel").Cr
eateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaObject.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaObject object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaObject("Period
ic").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaObject.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java object. If the object exists a message box appears confirming its existence.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaObject("username").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaObject.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyObject_0" JavaObject.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaObject("MyObject_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaObject.FireEventEx *JavaClassName*, *EventID* [, *constructor_param*$_3$,..., *contructor_param*$_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyObject_0" JavaObject.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaObject("MyObject_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaObject.GetTOProperty(*Property***, [, *Flags*, *Type*, *ParameterName*])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaObject name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaObject("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaObject.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaObject("Periodic"
).MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaObject.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaObject.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaObject("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaObject.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaObject.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaObject("puzzle.html").Set
TOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaObject.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaObject.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaObject("Panel").Ty
pe "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object
Set rect=MyObj.getBounds()
```

JavaRadioButton Object

A Java Radio Button.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- Set Method

- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaRadioButton.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaRadioButton object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaRadioButton("
Periodic").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaRadioButton.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaRadioButton("Pane
l").CreateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaRadioButton.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaRadioButton object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaRadioButton("
Periodic").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaRadioButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java radio button. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").
JavaRadioButton("username").Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaRadioButton.FireEvent(*constant* **[,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyRadioButton_0" JavaRadioButton.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaRadioButt
on("MyRadioButton_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4,
1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaRadioButton.FireEventEx *JavaClassName*, *EventID*
[, *constructor_param*$_3$,..., *contructor_param*$_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyRadioButton_0" JavaRadioButton.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaRadioButt
on("MyRadioButton_0").FireEventEx "java.awt.event.MouseEvent",
"MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaRadioButton.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaRadioButton name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaRadioButton("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1, y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaRadioButton.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaRadioButton("Periodic").MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaRadioButton.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaRadioButton.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaRadioButton("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

Set Method

Sets the value or state of the JavaRadioButton object.

Syntax

JavaRadioButton.Set

Example

The following example uses the **Set** method to select the radio button.

```
Browser("SwingSet demo").Page("SwingSet
demo_9").Applet("SwingSetApplet").JavaRadioButton("0").Set
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaRadioButton.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaRadioButton.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaRadioButton("puzzle.html
").SetTOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaRadioButton.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaRadioButton.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaRadioButton("Pane
l").Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

Set myObj=*JavaObjectName*(*ObjectName*).Object
Set myObj.*Method_to_activate*()

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaSlider Object

A Java Slider.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Drag Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaSlider.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaSlider object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaSlider("Periodi
c").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaSlider.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaSlider("Panel").Cre
ateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaSlider.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaSlider object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaSlider("Periodi
c").DblClick 503, 55, "LEFT"
```

Drag Method

Drags a slider or a scroll bar to a new position.

Syntax

JavaSlider.Drag *count*

Argument	Type	Description
count	variant	Indicates the position to which the slider or scroll bar is dragged.

Example

The following example uses the **Drag** method to drag the "player" slider in the music.html applet.

```
Browser("Music").Page("Music").Applet("music.html").JavaSlider("Player").Drag
"11"
```

Exist Method

Checks that an object exists.

Syntax

JavaSlider.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java slider. If the object exists a message box appears confirming its existence.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaSlider("username").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaSlider.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MySlider_0" JavaSlider.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaSlider("My
Slider_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaSlider.FireEventEx *JavaClassName*, *EventID* [, $constructor\_param_3$,..., $contructor\_param_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MySlider_0" JavaSlider.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaSlider("My
Slider_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0,
"BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaSlider.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaSlider name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaSlider("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x1,y1 coordinates to the specified x2, y2 coordinates.

Syntax

JavaSlider.MouseDrag x_1, y_1, x_2, y_2, [*button*]

Argument	Type	Description
x_1, y_1	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, y_2	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaSlider("Periodic").
MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaSlider.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaSlider.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaSlider("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaSlider.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaSlider.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaSlider("puzzle.html").SetT
OProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaSlider.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaSlider.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaSlider("Panel").
Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaSpin Object

A Java Spin object.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaSpin.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaSpin object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaSpin("Periodic"
).Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaSpin.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaSpin("Panel").Creat
eObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaSpin.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaSpin object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaSpin("Periodic"
).DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaSpin.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "spinner" Java spin. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaSpin("spinner").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaSpin.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MySpinButton_0" JavaSpin.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaSpin("MySpinButton_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaSpin.FireEventEx *JavaClassName*, *EventID* [, *constructor_param*$_3$,..., *contructor_param*$_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MySpinButton_0" JavaSpin.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaSpin("MySpinButton_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaSpin.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaSpin name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaSpin("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaSpin.MouseDrag *x_1*, *y_1*, *x_2*, *y_2*, [*button*]

Argument	Type	Description
x_1, *y_1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, *y_2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html")JavaSpin("Periodic").MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaSpin.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaSpin.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaSpin("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

Set Method

Sets the value of the JavaSpin object.

Syntax

JavaSpin.Set *SetValue*

Argument	Type	Description
SetValue	variant	The value or state to be assigned to the object.

Example

The following example uses the **Set** method to set the value of the Identical spin JavaSpin object.

```
Browser("SwingSet
demo").Page("Spin").Applet("SpinApplet").JavaSpin("Identical spin").Set
"Identical item"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaSpin.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaSpin.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaSpin("puzzle.html").SetT
OProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaSpin.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaSpin.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaSpin("Panel").Type
"ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

Set myObj=*JavaObjectName*(*ObjectName*).Object
Set myObj.*Method_to_activate*()

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaTab Object

A Java tab.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetTOProperty Method
- MouseDrag Method
- GetROProperty Method
- SetTOProperty Method
- Select Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaTab.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the periodic.html applet.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaTab("Periodic")
.Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaTab.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaTab("Panel").Creat
eObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaTab.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaTab object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaTab("Periodic")
.DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaTab.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "methods" Java tab. If the object exists a message box appears confirming its existence.

```
If Browser("Browser").Page("Page").Applet("login.html").JavaTab("methods").
Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaTab.FireEvent(*constant* [**,** *EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyTab_0" JavaTab.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaTab("MyTab_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaTab.FireEventEx *JavaClassName*, *EventID* [, *constructor_param*$_3$,..., *contructor_param*$_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyTab_0" JavaTab.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaTab("MyTab_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaTab.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaTab name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaTab("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaTab.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaTab("Periodic").MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaTab.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaTab.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaTab("puzzle.html").GetROProperty ("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaTab.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaTab.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaTab("puzzle.html").SetTO
Property "Name", "NewName", "Long", "", "1"
```

Select Method

Selects the specified tab panel.

Syntax

JavaTab.Select *TabName*

Argument	Type	Description
TabName	variant	The name (label) of the tab panel to select.

Example

The following example selects the "Menus & ToolBars" tab in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTab("JTabbedPane").Select "Menus
& ToolBars"
```

Type Method

Types the specified text in the object.

Syntax

JavaTab.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaTab.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaTab("Panel").Type
"ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

JavaObjectName(*ObjectName*).Object.*Method_to_activate*()
or

Set myObj=*JavaObjectName*(*ObjectName*).Object
Set myObj.*Method_to_activate*()

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaTable Object

A Java table.

Associated Methods:

- ActivateCell Method
- ActivateColumn Method
- ActivateRow Method
- Click Method
- ClickCell Method
- CreateObject Method
- DblClick Method
- DeselectCell Method
- DeselectColumn Method
- DeselectColumnsRange Method
- DeselectRow Method
- DeselectRowsRange Method
- DoubleClickCell Method
- Drag Method
- Exist Method
- ExtendColumn Method
- ExtendColumnsRange Method
- ExtendRow Method
- ExtendRowsRange Method
- FireEvent Method
- FireEventEx Method
- GetCellData Method
- GetTOProperty Method

- MouseDrag Method
- GetROProperty Method
- SelectCell Method
- SelectCellsRange Method
- SelectColumn Method
- SelectColumnHeader Method
- SelectColumnsRange Method
- SelectRow Method
- SelectRowsRange Method
- SetCellData Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

ActivateCell Method

Activates the specified cell in the table.

Syntax

JavaTable.ActivateCell *row*, *col*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.
col	variant	Indicates the column number or the column header label.

Example

The following example activates the cell in row 1, column 3 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").ActivateCell
1, 3
```

ActivateColumn Method

Activates the specified column in the table.

Syntax

JavaTable.ActivateColumn *col*

Argument	Type	Description
col	variant	Indicates the column number or the column header label.

Example

The following example activates column 2 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").ActivateColumn 2
```

In the following example, the column header called "employee names" is activated.

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").ActivateColumn "employee names"
```

ActivateRow Method

Activates the specified row in the table.

Syntax

JavaTable.ActivateColumn *row*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.

Example

The following example activates row 1 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").ActivateRow
1
```

In the following example, the row header called "amount paid" is activated.

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").ActivateRow
"amount paid"
```

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaTable.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the periodic.html applet.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaTable("Periodi
c").Click 503, 55, "LEFT"
```

ClickCell Method

Clicks the specified cell in the table.

Syntax

JavaTable.ClickCell *row*, *col* [,*button*, *modifier(s)*]

Argument	Type	Description
row	String	Indicates the row number or the row header label.
col	String	Indicates the column number or the column header label.

Argument	Type	Description
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**
modifier(s)	String	**Optional.** Indicates if SHIFT and/or CONTROL keys were involved in this action. **Default="NONE"**

Example

The following example performs a shift-click the cell in row 2 of the “Last Name” column in the SwingSet Applet JavaTable:

```
Browser("SwingSet demo").Page("SwingSet
demo_13").Applet("SwingSetApplet").JavaTable("Inter-cell spacing:").ClickCell
2, "Last Name", "LEFT", "SHIFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaTable.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaTable("Panel").Cre
ateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaTable.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaTable object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaTable("Periodi
c").DblClick 503, 55, "LEFT"
```

DeselectCell Method

Deselects the specified cell in the table.

Syntax

JavaTable.DeselectCell *row*, *col*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.
col	variant	Indicates the column number or the column header label.

Example

The following example deselects the cell in row 1, column 2 (labeled "Favorite color") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").DeselectCell
1, "Favorite color"
```

DeselectColumn Method

Deselects the specified column in the table.

Syntax

JavaTable.DeselectColumn *col*

Argument	Type	Description
col	variant	Indicates the column number or the column header label.

Example

The following example deselects column 2 (labeled "Favorite color") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").DeselectColumn "Favorite color"
```

DeselectColumnsRange Method

Deselects the specified columns in the table.

Syntax

JavaTable.DeselectColumnsRange *start_col*, *end_col*

Argument	Type	Description
start_col	variant	Indicates the starting row number or the row header label.
end_col	variant	Indicates the ending column number or the column header label.

Example

The following example deselects column 2 (labeled "Favorite color") through column 5 (labeled "Favorite song") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").DeselectColumnsRange "Favorite color", "Favorite song"
```

DeselectRow Method

Deselects the specified row in the table.

Syntax

JavaTable.DeselectRow *row*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.

Example

The following example deselects row 1 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").DeselectRow
1
```

DeselectRowsRange Method

Deselects the specified rows in the table.

Syntax

JavaTable.DeselectRowsRange *start_row*, *end_row*

Argument	Type	Description
start_row	variant	Indicates the starting row number or the row header label.
end_row	variant	Indicates the ending row number or the row header label.

Example

The following example Deselects row 1 through row 3 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").DeselectRowsRange 1, 3
```

DoubleClickCell Method

Double-clicks the specified cell in the table.

Syntax

JavaTable.DoubleClickCell *row*, *col* [, *button*, *modifier1*, *modifier2*]

Argument	Type	Description
row	String	Indicates the row number or the row header label.
col	String	Indicates the column number or the column header label.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**
modifier(s)	String	**Optional.** Indicates if SHIFT and/or CONTROL keys were involved in this action.

Example

The following example double-clicks the cell in row 4, column 3 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").DoubleClickCell 4, 3
```

Drag Method

Performs a mouse drag from the indicated starting cell to the indicated ending cell.

Syntax

JavaTable.Drag *start_row*, *start_col*, *end_row*, *end_column* [, *button*, *modifier(s)*]

Argument	Type	Description
start_row	variant	Indicates the starting row number or the row header label.
start_col	variant	Indicates the starting column number or the column header label.
end_row	variant	Indicates the ending row number or the row header label.
start_col	variant	Indicates the ending column number or the column header label.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**
modifier(s)	String	**Optional.** Indicates if SHIFT and/or CONTROL keys were involved in this action. **Default = "NONE"**

Example

The following example uses the **Drag** method to drag the mouse from row 1, column 1 to row 4, column 3 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").Drag 1, 1, 4,
3
```

Exist Method

Checks that an object exists.

Syntax

JavaTable.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java table. If the object exists a message box appears confirming its existence.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaTable("username").
Exist Then
msgbox("The object exists.")
```

ExtendColumn Method

Selects an additional column in the table.

Syntax

JavaTable.ExtendColumn *col*

Argument	Type	Description
col	variant	Indicates the column number or the column header label to select.

Example

The following example adds column 2 to the column or columns that are already selected in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").ExtendColumn 2
```

ExtendColumnsRange Method

Selects the specified range of columns in the table.

Syntax

JavaTable.ExtendColumnsRange *start_col*, *end_col*

Argument	Type	Description
start_col	variant	Indicates the starting column number or the row header label.
end_col	variant	Indicates the ending column number or the row header label.

Example

The following example adds columns 2-4 to the column or columns that are already selected in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").ExtendColumnsRange 2, 4
```

ExtendRow Method

Selects an additional row in the table.

Syntax

JavaTable.ExtendRow row

Argument	Type	Description
Row	variant	Indicates the row number or the row header label.

Example

The following example adds row 3 to the row or rows that are already selected in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").ExtendRow 3
```

ExtendRowsRange Method

Selects the specified range of rows in the table.

Syntax

JavaTable.ExtendRowsRange *start_row*, *end_row*

Argument	Type	Description
start_row	variant	Indicates the starting row number or the row header label.
end_row	variant	Indicates the ending row number or the row header label.

Example

The following example adds rows 2-4 to the row or rows that are already selected in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").ExtendRowsRange 2, 4
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaTable.FireEvent(*constant* **[,** *EventParam(s)*] **)**

Argument	Type	Description
Constant	FireEventType	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	variant	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyTable_0" JavaTable.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaTable("My
Table_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaTable.FireEventEx *JavaClassName*, *EventID* [, $constructor\_param_3$,..., $contructor\_param_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	variant	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyTable_0" JavaTable.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaTable("My
Table_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED", 0,
"BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetCellData Method

Returns the data contained in the specified cell.

Syntax

GetCellData(*row*, *column* **)**

Argument	Type	Description
row	variant	Indicates the row number or the row header label.
col	variant	Indicates the column number or the column header label.

Return Value

Variant

Example

The following example retrieves the data contained in the cell in the first row and first column.

```
Browser("SwingSet demo").Page("SwingSet
demo_13").Applet("SwingSetApplet").JavaTable("Inter-cell
spacing:").GetCellData 1, 1
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaTable.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the JavaTable name.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").Applet("MercuryLogo").JavaTable("Mercury").
GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaTable.MouseDrag *x_1*, *y_1*, *x_2*, *y_2*, [*button*]

Argument	Type	Description
x_1, *y_1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x_2, *y_2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaTable("Periodic").
MouseDrag 4, 10, 10, 10, "LEFT"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaTable.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaTable.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaTable("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

SelectCell Method

Selects the specified cell in the table.

Syntax

JavaTable.SelectCell *row*, *col*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.
col	variant	Indicates the column number or the column header label.

Example

The following example selects the cell in row 1, column 2 (labeled "Favorite color") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").SelectCell 1,
"Favorite color"
```

SelectCellsRange Method

Selects the specified cells in the table.

Syntax

JavaTable.SelectCellsRange *start_row*, *start_col*, *end_row*, *end_col*

Argument	Type	Description
start_row	variant	Indicates the starting row number or the row header label.
start_col	variant	Indicates the starting column number or the column header label.
end_row	variant	Indicates the ending row number or the row header label.
end_col	variant	Indicates the ending column number or the column header label.

Example

The following example selects the cells in row 1, column 2 (labeled "Favorite color") through row 3, column 5 (labeled "Favorite song") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").SelectCellsRange 1, "Favorite color" , 3, "Favorite song"
```

SelectColumn Method

Selects the specified column in the table.

Syntax

JavaTable.SelectColumn *col*

Argument	Type	Description
col	variant	Indicates the column number or the column header label.

Example

The following example selects column 2 (labeled "Favorite color") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").SelectColumn "Favorite color"
```

SelectColumnHeader Method

Selects the specified column header in the table.

Syntax

JavaTable.SelectColumnHeader *Col* [**,** *MouseButton*]

Argument	Type	Description
Col	variant	Indicates the column number or the column header label
MouseButton	string	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the right mouse button to select the First Name column header.

```
Browser("SwingSet demo").Page("SwingSet
demo_13").Applet("SwingSetApplet").JavaTable("Inter-cell
spacing:").SelectColumnHeader "First Name", "RIGHT"
```

SelectColumnsRange Method

Selects the specified columns in the table.

Syntax

JavaTable.SelectColumnsRange *start_col*, *end_col*

Argument	Type	Description
start_col	variant	Indicates the starting column number or the column header label.
end_col	variant	Indicates the ending column number or the column header label.

Example

The following example selects column 2 (labeled "Favorite color") through column 5 (labeled "Favorite song") in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").SelectColumnsRange "Favorite color", "Favorite song"
```

SelectRow Method

Selects the specified row in the table.

Syntax

JavaTable.SelectRow *row*

Argument	Type	Description
row	variant	Indicates the row number or the row header label.

Example

The following example selects row 1 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").SelectRow 1
```

SelectRowsRange Method

Selects the specified rows in the table.

Syntax

JavaTable.SelectRowsRange *start_row*, *end_row*

Argument	Type	Description
start_row	variant	Indicates the starting row number or the row header label.
end_row	variant	Indicates the ending row number or the row header label.

Example

The following example selects row 1 through row 3 in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table
Example").SelectRowsRange 1, 3
```

SetCellData Method

Sets the cell contents with the specified data.

Syntax

JavaTable.SetCellData *row*, *col*, *data*

Argument	Type	Description
Row	variant	Indicates the row number or the row header label.
Col	variant	Indicates the column number or the column header label.
Data	string	Indicates the data to be inserted in the specified cell.

Example

The following example sets the cell contents in the 2nd row and 4rd column to "2 passengers" in the "Table Example" JavaTable in the SwingSet Applet:

```
Browser("Browser").Page("SwingSet
demo").Applet("swingsetapplet.html").JavaTable("Table Example").SetCellData
2, 4, "2 passengers"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaTable.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaTable.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaTable("puzzle.html").SetT
OProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaTable.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaTable.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaTable("Panel").Type "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaObjectName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaObjectName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

JavaToolBar Object

A Java tool bar.

Associated Methods:

- Click Method
- CreateObject Method
- DblClick Method
- Exist Method
- FireEvent Method
- FireEventEx Method
- GetROProperty Method
- GetTOProperty Method
- MouseDrag Method
- Press Method
- SetTOProperty Method
- Type Method

Associated Properties:

- Object Property

Click Method

Clicks the specified location with the specified mouse button.

Syntax

JavaToolBar.Click *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **Click** method to click on the Periodic JavaToolBar object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaToolBar("Periodic").Click 503, 55, "LEFT"
```

CreateObject Method

Creates an instance of any Java object within your application.

Syntax

JavaToolBar.CreateObject(*ClassName*, [$consArg_1$, ... , $consArg_X$] **);**

Argument	Type	Description
ClassName	string	The Java class name.
$consArg_1$... $consArg_X$	variant	Required event arguments for the object constructor.

Example

The following example uses the **CreateObject** method to create a rectangle object.

```
Set Rect =
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaToolBar("Panel").Cr
eateObject ("java.awt.Rectangle", 10, 20)
```

DblClick Method

Double-clicks the specified location with the specified mouse button.

Syntax

JavaToolBar.DblClick *x*, *y*, [*button*]

Argument	Type	Description
x, y	long	Indicates the x- and y-coordinates of the location on the applet to be clicked.
button	String	**Optional.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button.

Example

The following example uses the **DblClick** method to double-click on the Periodic JavaToolBar object.

```
Browser("Periodic").Page("Periodic").Applet("periodic.html").JavaToolBar("Perio
dic").DblClick 503, 55, "LEFT"
```

Exist Method

Checks that an object exists.

Syntax

JavaToolBar.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" Java tool bar. If the object exists a message box appears confirming its existence.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaToolBar("username")
.Exist Then
msgbox("The object exists.")
```

FireEvent Method

Simulates an event on a Java object using one of several pre-defined event constants.

Note: If the list of pre-defined constants does not cover the event you want to fire, you can use the FireEventEx method to fire any event that has a constructor.

Syntax

JavaToolBar.FireEvent(*constant*, [*EventParam(s)*] **)**

Argument	Type	Description
constant	long	The name of the pre-defined constant for the event you want to fire. The available constants are: micMouseEnter, micMouseExit, micMouseClick, micMousePress, micMouseRelease, micMouseDrag, micMouseMove, micKeyPress, micKeyRelease, micKeyType, micFocusGain, micFocusLost
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEvent** method to fire a mouseClick event on the "MyToolbar_0" JavaToolBar.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaToolBar("
MyToolbar_0").FireEvent micMouseClick, 0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEventEx method to fire this event.

FireEventEx Method

Simulates an event on a Java object, given the ClassName, Event ID and Event Parameters.

Note: You can use FireEventEx for any Java event.

Syntax

JavaToolBar.FireEventEx *JavaClassName*, *EventID* [, $constructor\_param_3$,..., $contructor\_param_X$]

Argument	Type	Description
JavaClassName	string	The name of the Java class representing the event to be activated.
EventID	string	The event ID number or the final field string that represents the event ID.
EventParam(s)	string	**If applicable.** All parameters of the constructor except for the source and EventID.

Example

The following example uses the **FireEventEx** method to fire a mouseClick event on the "MyToolbar_0" JavaToolBar.

```
Browser("Browser").Page("Page").Applet("mybuttonapplet.htm").JavaToolBar("
MyToolbar_0").FireEventEx "java.awt.event.MouseEvent", "MOUSE_CLICKED",
0, "BUTTON1_MASK", 4, 4, 1, "OFF"
```

Note: You can also use the FireEvent method to fire this event.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

JavaToolBar.GetROProperty(*Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the "Value" attribute of the JavaToolBar.

```
CheckState = Browser.Page("Table
Sample").Applet("puzzle.html").JavaToolBar("puzzle.html").GetROProperty
("Value") ' CheckState contains "OFF"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

JavaToolBar.GetTOProperty(*Property* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.

Argument	Type	Description
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the HTML name of the tool bar.

```
Dim ObjectName
ObjectName = Browser("Browser Name").Page("Mercury
Interactive").JavaToolBar("MercuryToolbar").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseDrag Method

Performs a mouse drag operation from the specified x_1,y_1 coordinates to the specified x_2, y_2 coordinates.

Syntax

JavaToolBar.MouseDrag *x1*, *y1*, *x2*, *y2*, [*button*]

Argument	Type	Description
x1, *y1*	long	Indicates the x- and y-coordinates of the location from which to begin the mouse drag.
x2, *y2*	long	Indicates the x- and y-coordinates of the location where the mouse drag stops.
button	String	**Not always required.** Indicates whether the click should be performed with the "LEFT" or "RIGHT" mouse button. **Default="LEFT".**

Example

The following example uses the **MouseDrag** method to move an object on the periodic.html applet.

```
Browser("Browser").Page("Page").Applet("periodic.html").JavaToolBar("Periodic
").MouseDrag 4, 10, 10, 10, "LEFT"
```

Press Method

Presses the specified tool bar button.

Syntax

JavaToolBar.Press *Button*

Argument	Type	Description
Button	String	The tool bar button name or its internal index.

Example

The following example presses the Help tool bar button.

```
Browser("Developer Server").Page("Developer
Server").Applet("Main").JavaToolBar("ToolBar").Press "Help"
```

The following example presses the tool bar button with the index: 22.

```
Browser("Developer Server").Page("Developer
Server").Applet("Main").JavaToolBar("ToolBar").Press 22
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

JavaToolBar.SetTOProperty *Property*, *Value* [, *Flags*, *Type*, *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the **SetTOProperty** method to set the name of a JavaToolBar.

```
Browser("Mercury
Tours").Page("FindFlights").Applet("puzzle.html").JavaToolBar("puzzle.html").Se
tTOProperty "Name", "NewName", "Long", "", "1"
```

Type Method

Types the specified text in the object.

Syntax

JavaToolBar.Type *Text*

Argument	Type	Description
Text	string	The text to type.

Example

The following example uses the **Type** method to type "ABC" in the JavaToolBar.

```
Browser("Periodic").Page("Periodic").Applet("Periodic").JavaToolBar("Panel").Ty
pe "ABC"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

*JavaToolBarName***(***ObjectName***).Object.***Method_to_activate***()**
or

Set myObj=*JavaToolBarName***(***ObjectName***).Object**
Set myObj.*Method_to_activate***()**

Example

In the following example, suppose the **getBounds** method is supported for your "Enter" JavaButton. To activate the **getBounds** method, you insert the following statement into your test script:

```
Set rect=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object.getBounds()
```

Alternatively, you could insert the following:

```
Set MyObj=Browser("Browser").Page("PushButtonDemo").
Applet("pushbutton.html").JavaButton("Enter").Object

Set rect=MyObj.getBounds()
```

4

Parameterization

The following objects can be used to perform parameterization-related operations.

- DataTable Object
- Environment Object
- Parameter Object
- RandomNumber Object
- Sheet Object

DataTable Object

The data table.

Associated Methods:

- AddSheet Method
- DeleteSheet Method
- Export Method
- GetCurrentRow Method
- GetRowCount Method
- GetSheet Method
- GetSheetCount Method
- Import Method
- SetCurrentRow Method

- SetFileName Method
- SetNextRow Method
- SetPrevRow Method

Associated Properties:

- GlobalSheet Property
- LocalSheet Property
- RawValue Property
- Value Property

AddSheet Method

Adds the specified sheet so that you can directly set properties of the new sheet in the same statement.

Syntax

DataTable.AddSheet(*SheetName* **)**

Argument	Type	Description
SheetName	String	Assigns a name to the new sheet.

Example

The following example uses the **AddSheet** method to create the new sheet, "MySheet". It is possible to use methods or check properties of the new sheet within the same statement.

```
DataTable.AddSheet("MySheet")
```

DeleteSheet Method

Deletes the specified sheet.

Syntax

DataTable.DeleteSheet *SheetID*

Argument	Type	Description
SheetID	Variant	Identifies the sheet to be returned. The SheetID can be the sheet name, index or dtLocalSheet. (Index values begin with 1)

Example

The following example uses the **DeleteSheet** method to delete the sheet, "MySheet".

```
DataTable.DeleteSheet "MySheet"
```

Export Method

Saves a copy of the data table in the specified location

Syntax

DataTable.Export(*Filename* **)**

Argument	Type	Description
Filename	String	The full path of the location to which the data table should be exported.

Example

The following example uses the **Export** method to save a copy of the test's data table in *C:\flights.xls*.

```
DataTable.Export ("C:\flights.xls")
```

GetCurrentRow Method

Returns the current (active) row in the global data sheet.

Syntax

DataTable.GetCurrentRow

Example

The following example uses the **GetCurrentRow** method to retrieve the row currently being used by the DataTable and writes it to the report.

```
row = DataTable.GetCurrentRow
Reporter.ReportEvent 1, "Row Number", row
```

GetRowCount Method

Returns the total number of rows in the longest column in the global data sheet or in the specified data sheet.

Syntax

DataTable.GetRowCount

Return Value

Number

Example

The following example uses the **GetRowCount** method to find the total number of rows in the longest column of the MySheet sheet and writes it to the report.

```
rowcount = DataTable.GetSheet("MySheet").GetRowCount
Reporter.ReportEvent 1, "There are ", rowcount, "rows in the data sheet."
```

GetSheet Method

Returns the specified sheet from the data table.

Syntax

DataTable.GetSheet(*SheetID* **)**

Argument	Type	Description
SheetID	variant	Identifies the sheet to be returned. The SheetID can be the sheet name, index or dtLocalSheet, or dtGlobalSheet. (Index values begin with 1)

Example

The following example uses the **GetSheet** method to return the "MySheet" sheet in order to add a parameter to it.

```
MyParam=DataTable.GetSheet ("MySheet").AddParameter("Time", "8:00")
```

You can also use this to add a parameter to the "MySheet" local sheet (note that no value is returned). Thus,

```
DataTable.GetSheet ("MySheet").AddParameter "Time", "8:00"
```

GetSheetCount Method

GetSheetCount Method

returns the total number of sheets in the data table.

Syntax

DataTable.GetSheetCount

Return Value

Number

Example

The following example uses the **GetSheetCount** method to find the total number of sheets in the test's data table and writes it to the report.

```
sheetcount = DataTable.GetSheetCount
Reporter.ReportEvent 1, "There are", sheetcount, "sheets in the data table."
```

Import Method

Imports the specified Excel file.

Syntax

DataTable.Import(*FileName* **)**

Argument	Type	Description
Filename	String	The full path of the excel table to import.

Example

The imported table replaces all data in the existing data table (including all data sheets).

Note: The imported table must match the test. The column names must match the parameters in the test, and the sheet names must match the action names.

The following example uses the **Import** method to import the *flights.xls* table to the data table.

```
DataTable.Import ("C:\flights.xls")
```

SetCurrentRow Method

Sets the specified row as the current (active) row.

Syntax

DataTable.SetCurrentRow(*RowNumber* **)**

Argument	Type	Description
RowNumber	Number	Indicates the number of the row to set as the active row (values begin with 1)

Example

The following example uses the **SetCurrentRow** method to change the active row to the second row in the global data table.

```
DataTable.SetCurrentRow (2)
```

SetFileName Method

Specifies the filename of the DataTable to use.

Syntax

DataTable.SetFileName *FileName*

Argument	Type	Description
FileName	String	Full pathname of the file to use as the DataTable. The file must have an .xls extension

Example

The following illustrates the use of the **SetFileName** method to change the name of the default DataTable.

```
DataTable.SetFileName "C:\DataFiles\Names.xls"
```

SetNextRow Method

Sets the row after the current (active) row as the new current row in the data table.

Note: If the current row is the last row in the data table, applying this method sets the first row in the data table as the new current row.

Syntax

DataTable.SetNextRow

Example

The following example uses the **SetNextRow** method to change the active row to the next row in the global data table.

```
DataTable.SetNextRow
```

SetPrevRow Method

Sets the row above the current (active) row as the new current (active) row.

Note: If the current row is the first row in the data table, applying this method sets the last row in the data table as the new current row.

Syntax

DataTable.SetPrevRow

Example

The following example uses the **SetPrevRow** method to change the active row to the previous row in the global data table.

```
DataTable.SetPrevRow
```

GlobalSheet Property

Returns the Global sheet.

Syntax

DataTable.GlobalSheet

Example

The following example uses the **GlobalSheet** property to return the global sheet in order to add a parameter (column) to it.

```
ParamValue=DataTable.GlobalSheet.AddParameter("Time", "5:45")
```

You can also use this to add a parameter to the global sheet (note that no value is returned):

```
DataTable.GlobalSheet.AddParameter "Time", "5:45"
```

LocalSheet Property

Returns the current (active) local sheet.

Syntax

DataTable.LocalSheet

Example

The following example uses the **LocalSheet** property to return the local sheet in order to add a parameter (column) to it.

```
MyParam=DataTable.LocalSheet.AddParameter("Time", "5:45")
```

RawValue Property

Retrieves the *raw value* of the cell in the specified parameter and the current row.

The *raw value* is the actual string written in a cell before the cell has been computed, such as the actual text from a formula.

Syntax

DataTable.RawValue *ParameterID* [*, SheetID*]

Argument	Type	Description
ParameterID	Variant	Identifies the parameter (column) of the value to be set/retrieved. (Index values begin with 1)
SheetID	Variant	**Optional.** Identifies the sheet to be returned. The SheetID can be the sheet name, index or dtLocalSheet, or dtGlobalSheet. If no Sheet is specified, the Global Sheet is used. (Index values begin with 1)

Example

The following example uses the RawValue property to find the formula used in the current row of the Date column in the ActionA sheet. The statement below returns the value: =NOW()

```
FormulaVal=DataTable.RawValue ("Date", "ActionA")
```

Value Property

DataTable default property.

Retrieves or sets the value of the cell in the specified parameter and the current row.

Note: This method returns the computed value of the cell. For example, if the cell contains a formula, the method returns TRUE or FALSE.

Syntax

To find the value:

DataTable.Value(*ParameterID* [, *SheetID*]**)**
or

DataTable(*ParameterID* [, *SheetID*]**)**

To set the value:

DataTable.Value(*ParameterID* [, *SheetID*]**)**=*newvalue*

or

DataTable(*ParameterID* [, *SheetID*]**)** =*newvalue*

Argument	Type	Description
ParameterID	Variant	Identifies the parameter (column) of the value to be set/retrieved. (Index values begin with 1.)
SheetID	Variant	**Optional.** Identifies the sheet to be returned. The SheetID can be the sheet name, index or dtLocalSheet, or dtGlobalSheet. If no Sheet is specified, the Global Sheet is used. (Index values begin with 1)
newvalue	String	Sets the value for the specified table cell.

Example

The following example uses the Value property to set the value in the current row of the Destination parameter (column) in the "ActionA" sheet.

```
DataTable.Value ("Destination", "ActionA")="New York"
```

The following example uses the Value property to set the value in the current row of the second parameter (column) in the third sheet.

```
DataTable.Value (2,3)="New York"
```

Note: You could omit the word Value in the statements above, because Value is the default property for the DataTable object.

The following example uses the default property to set the value in the current row of the Destination parameter (column) in the current (active) local sheet.

```
DataTable("Destination", dtlocalSheet)="New York"
```

Environment Object

Enables you to work with environment variables.

You can set or retrieve the value of environment variables using the Environment object.

Syntax

To set the value of a loaded environment variable, or to create and set the value of a new internal user-defined variable:

Environment(*VariableName* **) =** *value*

To retrieve the value of a loaded environment variable:

value = **Environment (** *VariableName* **)**

Argument	Type	Description
VariableName	String	The name of the environment variable.
Value	variant	The value of the environment variable.

Example:

The following example creates a new internal user-defined variable named MyVariable with a value of 10, and then retrieves the variable value and stores it in the MyValue variable.

```
Environment("MyVariable")=10
MyValue=Environment("MyVariable")
```

Associated Methods:

➤ LoadFromFile Method

LoadFromFile Method

Loads the specified environment variable file.

Syntax

Environment.LoadFromFile(*Path* **)**

Argument	Type	Description
Path	String	The path of the environment file to load.

Example:

The following example loads the MyVariables.txt file.

```
Environment.LoadFromFile(C:\QuickTest\Files\MyVariables.txt)
```

Parameter Object

A parameter (column) in a sheet in the data table.

Associated Properties:

- Name Property
- RawValue Property
- Value Property
- ValueByRow Property

Name Property

Returns the name of the parameter (column).

Syntax

Parameter.Name

Example

The following example uses the **Name** method to return the name of the newly created parameter writes it in the report.

```
Dim paramname
paramname = DataTable.LocalSheet.AddParameter("Food").Name
Reporter.ReportEvent 1, "The New Parameter name is", paramname
```

RawValue Property

Retrieves the *raw value* of the cell in the current row of the parameter.

The *raw value* is the actual string written in a cell before the cell has been computed, such as the actual text from a formula.

Syntax

Parameter.RawValue

Example

The following example uses the RawValue property to find the formula used in the current row of the Date column in the ActionA sheet. The statement below returns the value: =NOW()

```
FormulaVal=DataTable.GetSheet("ActionA").GetParameter("Date").RawValue
```

Value Property

Parameter default property.

Retrieves or sets the value of the cell in the current (active) row of the parameter.

Note: This method returns the computed value of the cell. For example, if the cell contains a formula, the method returns TRUE or FALSE.

Syntax

To find the value:

Parameter.Value or Parameter
To set the value:

Parameter.Value =*newvalue* or Parameter =*newvalue*

Example

The following example uses the Value property to set the value in the current row of the Destination parameter (column) in the "ActionA" sheet.

```
DataTable.GetSheet("ActionA").GetParameter("Destination").Value="New York"
```

Note: You could omit the word Value in the statement above, because Value is the default property for the Parameter object.

ValueByRow Property

Retrieves the value of the cell in the specified row of the parameter.

Syntax

Parameter.ValueByRow(*RowNum* **)**

Argument	Type	Description
RowNum	Number	Indicates the row of the parameter that should be returned. (Row numbers start with 1.)

Example

The following example uses the ValueByRow property to find the value in the 4th row of the Destination parameter (column) in the "ActionA" sheet.

```
DataTable.GetSheet("ActionA").GetParameter("Destination").ValueByRow(4)
```

RandomNumber Object

Enables you to work with random parameters.

You can generate a value for the specified random parameter using the RandomNumber object.

Syntax

To generate any random number:

RandomNumber(*ParameterName* **)**

Argument	Type	Description
ParameterName	String	The name of the RandomNumber parameter.

To generate a random within the specified range:

RandomNumber(*StartNumber, EndNumber***)**

Argument	Type	Description
StartNumber	Integer	The start number for the range within which the random number is generated.
EndNumber	Integer	The end number for the range within which the random number is generated.

Example:

The following example generates a random number between 0 and 100.

```
x=RandomNumber(0,100)
```

The following example generates a random number for the MyRandom parameter.

```
x=RandomNumber("MyRandom")
```

Associated Properties

➤ Value Property

Value Property

Generates a value for the specified RandomNumber parameter.

Syntax

To generate any random number:

RandomNumber.Value(*ParameterName* **)**

To generate a random within the specified range:

RandomNumber.Value(*StartNumber, EndNumber***)**

Argument	Type	Description
ParameterName	String	The name of the RandomNumber parameter.
StartNumber	Integer	The start number for the range within which the random number is generated.
EndNumber	Integer	The end number for the range within which the random number is generated.

Note: The Value property is the default property for the RandomNumber object. Thus you can also generate a random number as described in the RandomNumber Object syntax.

Example:

The following example generates a random number between 0 and 100.

```
x=RandomNumber.Value(0,100)
```

The following example generates a random number for the MyRandom parameter.

```
x=RandomNumber.Value("MyRandom")
```

Sheet Object

A sheet in the data table.

Associated Methods:

- AddParameter Method
- DeleteParameter Method
- GetCurrentRow Method
- GetParameter Method
- GetParameterCount Method
- GetRowCount Method
- SetCurrentRow Method
- SetNextRow Method
- SetPrevRow Method

Associated Properties:

- Name Property

AddParameter Method

Adds the specified parameter (column) to the sheet, sets the value of the first row to the specified value, and returns the parameter so that you can directly set or retrieve properties of the new parameter in the same statement.

Syntax

Sheet.AddParameter(*ParameterName, Value* **)**

Argument	Type	Description
ParamterName	String	Assigns a name to the new parameter. If another parameter in the sheet has the same name, a number (i.e. '1') will be appended to the assigned *ParameterName*. If the *ParameterName* contains illegal characters, the illegal characters will be replaced with '_' characters.
Value	String	Assigns a value to the first row of the parameter.

Example

The following example uses the **AddParameter** method to create the new Parameter, "Arrival" within the new sheet, MySheet, and sets the first cell in the column as "New York". Because the method also returns the newly created parameter, it is possible to use methods or check properties of the new sheet within the same statement.

```
ParamName=DataTable.AddSheet("MySheet").AddParameter("Arrival", "New York").Name
```

Note that if a parameter with the name "Arrival" already exists in the sheet, the example above will return "Arrival1" as the actual name assigned to the new parameter.

DeleteParameter Method

Deletes the specified parameter from the sheet.

Syntax

Sheet.DeleteParameter(*ParameterID* **)**

Argument	Type	Description
ParameterID	variant	Identifies the parameter (column) to be deleted by name or index. (Index values begin with 1)

Example

The following example uses the **DeleteParameter** method to delete the parameter, "Arrival" from the "MySheet" sheet.

```
DataTable.GetSheet("MySheet").DeleteParameter("Arrival")
```

Note that deleting a parameter from the sheet will cause the test to fail if a corresponding parameter exists in the test.

GetCurrentRow Method

Returns the current (active) row in the sheet.

Syntax

Sheet.GetCurrentRow

Example

The following example uses the **GetCurrentRow** method to retrieve the row currently being used by the DataTable and writes it to the report.

```
row = DataTable.GetSheet("MySheet").GetCurrentRow
Reporter.ReportEvent 1, "Row Number", row
```

GetParameter Method

Retrieves the specified parameter from the sheet.

Syntax

Sheet.GetParameter(*ParameterID* **)**

Argument	Type	Description
ParameterID	String/Index	Identifies the parameter (column) to be returned by name or index. (Index values begin with 1)

Example

The following example uses the **GetParameter** method to return the "Destination" parameter from the MySheet sheet.

```
DataTable.GetSheet("MySheet").GetParameter("Destination")
```

GetParameterCount Method

Returns the total number of parameters (columns) in the sheet.

Syntax

Sheet.GetParameterCount

Return Value

Number

Example

The following example uses the **GetParameterCount** method to find the total number of parameters (columns) in the MySheet sheet and writes it to the report.

```
paramcount = DataTable.GetSheet("MySheet").GetParameterCount
Reporter.ReportEvent 1, "There are ", paramcount, "columns in the data sheet."
```

GetRowCount Method

Returns the total number of rows in the longest column in the data sheet.

Syntax

Sheet.GetRowCount

Return Value

Number

Example

The following example uses the **GetRowCount** method to find the total number of rows in the first column of the MySheet sheet and writes it to the report.

```
rowcount = DataTable.GetSheet("MySheet").GetRowCount
Reporter.ReportEvent 1, "There are ", rowcount, "rows in the data sheet."
```

SetCurrentRow Method

Sets the specified row as the current (active) row.

Syntax

Sheet.SetCurrentRow(*RowNumber* **)**

Argument	Type	Description
RowNumber	Number	Indicates the number of the row to set as the active row

Example

The following example uses the **SetCurrentRow** method to change the active row to the second row in the MySheet data sheet.

```
DataTable.GetSheet("MySheet").SetCurrentRow(2)
```

SetNextRow Method

Sets the row after the current (active) row as the new current row in the data table.

Note: If the current row is the last row in the data table, applying this method sets the first row in the data table as the new current row.

Syntax

Sheet.SetNextRow

Example

The following example uses the **SetNextRow** method to change the active row to the next row data sheet.

```
DataTable.GetSheet("MySheet").SetNextRow
```

SetPrevRow Method

Sets the row above the current (active) row as the new current (active) row.

Syntax

Sheet.SetPrevRow

Example

The following example uses the **SetPrevRow** method to change the active row to the previous row in the data sheet.

```
DataTable.GetSheet("MySheet").SetPrevRow
```

Name Property

Returns the name of the data sheet.

Syntax

Sheet.Name

Example

The following example uses the **Name** method to return the name of the active sheet and writes it in the report.

```
Sheetname = DataTable.LocalSheet.Name
Reporter.ReportEvent 1, "The Active Sheet is", Sheetname
```

5

Real Player/Windows Media Player

The following objects can be used when testing Real Player or Windows Media Player applications or controls.

Note: Real Player and Media Player objects are supported in QuickTest Professional only.

- RealControl Object
- RealPlayer Object
- WMControl Object
- WMPlayer Object

RealControl Object

A Real control.

Associated Methods:

- GetCurrentTime Method
- MakeVisible Method
- Pause Method
- Play Method
- ProportionalClick Method
- Seek Method

- Stop Method
- WaitClipEnd Method
- WaitClipTime Method

Associated Properties:

- Object Property

GetCurrentTime Method

Retrieves the current time position of a Real object in seconds.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.GetCurrentTime()

Return Value

Number

Example

The following example uses the **GetCurrentTime** method to retrieve the current position of the Top Stories clip.

```
time = RealControl("Top Stories").GetCurrentTime()
```

MakeVisible Method

If the RealControl object is not visible in the window, the **MakeVisible** method scrolls it into view.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.MakeVisible

Example

The following example brings the CIRC3 RealControl into view.

```
Browser("Welcome to A-Soft").Page("Untitled
Normal").RealControl("CIRC3.Circ3Ctrl.1").MakeVisible
```

Pause Method

Pauses playing at the current location (time position).

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.Pause

Example

The following example uses the **Pause** method to pause the Top Stories clip.

```
RealPlayer("Top Stories").Pause
```

Play Method

Starts or resumes playing the clip.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.Play

Example

The following example uses the **Play** method to resume play of the Top Stories clip.

```
RealRealControl("Top Stories").Play
```

ProportionalClick Method

Clicks on the Real Control or RealPlayer based on the relative location of the click proportional to the size of the entire screen.

Note that this method is applicable only on interactive video clips.

Syntax

RealControl.ProportionalClick(*PercentX, PercentY,* [*button*]**)**

Argument	Type	Description
PercentX, Percent Y	number	The relative coordinates of the click as a percentage of the width (X) and height(Y) of the screen.
button	integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example uses the **ProportionalClick** method to click on the clip in a location 2.20% into the screen, and 12.1% down from the top.

```
RealControl("Top Stories").ProportionalClick 2.20, 12.1
```

Seek Method

Jumps to the specified time position in the clip (in seconds).

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.Seek(*position_seconds***)**

Argument	Type	Description
position_seconds	number	the location in the clip to jump to (in seconds)

Example

The following example uses the **Seek** method to jump to 23.2 seconds into the Top Stories clip.

```
RealControl("Top Stories").Seek 23.2
```

Stop Method

Stops playing and removes clip from memory.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.Stop

Example

The following example uses the **Stop** method to stop play of the Top Stories clip.

```
RealControl("Top Stories").Stop
```

WaitClipEnd Method

Waits for the end of the clip.

Note: This method produces an error if the clip is stopped before the end by the user or an error in the application.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.WaitClipEnd

Example

The following example uses the **WaitClipEnd** method to suspend all further action until the clip reaches its end.

```
RealControl("Top Stories").WaitClipEnd
```

WaitClipTime Method

Waits for the clip to reach the specified time spot.

Note: This method is also supported for the **WMControl** object.

Syntax

RealControl.WaitClipTime(*ClipTime***)**

Argument	Type	Description
ClipTime	number	The location in the clip (in seconds from the start of the clip). The ClipTime is expressed to the nearest 100th of a second.

Example

The following example uses the **WaitClipTime** method to suspend all further action until the clip reaches 23.2 seconds.

```
RealControl("Top Stories").WaitClipTime 23.2
```

Note: The **WaitClipTime** method is automatically inserted when you record a **Stop** or **Pause**, in order to assure that the stop or pause occurs at the appropriate point.

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Note: This property is also supported for the **WMControl** object.

Syntax

EnvironmentObject(ObjectName).Object.Method_to_activate()

or

myObj=EnvironmentObject(ObjectName).Object
myObj.Method_to_activate()

Example

In the following example, suppose the **DoStop** method is supported for your Real control. To activate the **DoStop** method, you would insert the following statement into your test script:

```
RealControl("Name").Object.DoStop
```

Alternatively you could insert the following:

```
myObj = RealControl("Name").Object()
myObj.DoStop
```

RealPlayer Object

A RealPlayer application.

Associated Methods:

- Close Method
- GetCurrentTime Method
- OpenURL Method
- Pause Method
- Play Method
- ProportionalClick Method
- Seek Method
- Stop Method
- WaitClipEnd Method
- WaitClipTime Method

Associated Properties:

- Object Property

Close Method

Closes the RealPlayer application.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.Close

Example

The following example uses the **Close** method to close the RealPlayer application.

```
RealPlayer("Top Stories").Close
```

GetCurrentTime Method

Retrieves the current time posiyion of a Real object in seconds.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.GetCurrentTime()

Return Value

Float

Example

The following example uses the **GetCurrentTime** method to retrieve the current position of the Top Stories clip.

```
time = RealPlayer("Top Stories").GetCurrentTime()
```

OpenURL Method

Opens the URL containing the Real clip.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.OpenURL(*movie_location* **)**

Argument	Type	Description
movie_location	string	the location of the movie.

Example

The following example uses the **OpenURL** method to open the Top Stories clip and begin playing it.

```
RealPlayer("Top Stories").OpenURL
"http://channels.real.com/vram/single?programs=98&tcode=98"
```

Pause Method

Pauses playing at the current location (time position).

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.Pause

Example

The following example uses the **Pause** method to pause the Top Stories clip.

```
RealPlayer("Top Stories").Pause
```

Play Method

Starts or resumes playing the clip.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.Play

Example

The following example uses the **Play** method to resume play of the Top Stories clip.

```
RealPlayer("Top Stories").Play
```

ProportionalClick Method

Clicks on the Real Control or RealPlayer based on the relative location of the click proportional to the size of the entire screen.

Note that this method is applicable only on interactive video clips.

Syntax

RealPlayer.ProportionalClick(*PercentX, PercentY,* [*button*]**)**

Argument	Type	Description
Percent X, Percent Y	number	The relative coordinates of the click as a percentage of the width (X) and height(Y) of thescreen.
button	integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example uses the **ProportionalClick** method to click on the clip in a location 2.20% into the screen, and 12.1% down from the top.

```
RealPlayer("Top Stories").ProportionalClick 2.20, 12.1
```

Seek Method

Jumps to the specified time position in the clip (in seconds).

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.Seek(*position_seconds***)**

Argument	Type	Description
position_seconds	number	the location in the clip to jump to (in seconds)

Example

The following example uses the **Seek** method to jump to 23.2 seconds into the Top Stories clip.

```
RealPlayer("Top Stories").Seek 23.2
```

Stop Method

Stops playing and removes clip from memory.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.Stop

Example

The following example uses the **Stop** method to stop play of the Top Stories clip.

```
RealPlayer("Top Stories").Stop
```

WaitClipEnd Method

Waits for the end of the clip.

Note: This method produces an error if the clip is stopped before the end by the user or an error in the application.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.WaitClipEnd

Example

The following example uses the **WaitClipEnd** method to suspend all further action until the clip reaches its end.

```
RealPlayer("Top Stories").WaitClipEnd
```

WaitClipTime Method

Waits for the clip to reach the specified time spot.

Note: This method is also supported for the **WMPlayer** object.

Syntax

RealPlayer.WaitClipTime(*ClipTime***)**

Argument	Type	Description
ClipTime	number	The location in the clip (in seconds from the start of the clip). The ClipTime is expressed to the nearest 100th of a second.

Example

The following example uses the **WaitClipTime** method to suspend all further action until the clip reaches 23.2 seconds.

```
RealPlayer("Top Stories").WaitClipTime 23.2
```

Note: The **WaitClipTime** method is automatically inserted when you record a **Stop** or **Pause**, in order to assure that the stop or pause occurs at the appropriate point.

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Note: This property is also supported for the **WMPlayer** object.

Syntax

EnvironmentObject(ObjectName).Object.Method_to_activate()
or

myObj=EnvironmentObject(ObjectName).Object
myObj.Method_to_activate()

Example

In the following example, suppose the **DoStop** method is supported for your Real control. To activate the **DoStop** method, you would insert the following statement into your test script:

```
RealPlayer("Name").Object.DoStop
```

Alternatively you could insert the following:

```
myObj = RealPlayer("Name").Object()
myObj.DoStop
```

WMControl Object

A Windows Media Player control.

Associated Methods:

- GetCurrentTime Method, see page 260
- MakeVisible Method, see page 261
- Pause Method, see page 261
- Play Method, see page 262
- Seek Method, see page 263
- Stop Method, see page 263
- WaitClipEnd Method, see page 264
- WaitClipTime Method, see page 265

Associated Properties:

- Object Property, see page 266

WMPlayer Object

A Windows Media Player application.

Associated Methods:

- Close Method, see page 267
- GetCurrentTime Method, see page 268
- OpenURL Method, see page 268
- Pause Method, see page 269
- Play Method, see page 270
- Seek Method, see page 271
- Stop Method, see page 271

- WaitClipEnd Method, see page 272
- WaitClipTime Method, see page 273

Associated Properties:

- Object Property, see page 274

6

Standard

The following objects can be used when testing standard Windows objects.

- Window Object
- Dialog Object
- WinButton Object
- WinCheckBox Object
- WinComboBox Object
- WinContextMenu Object
- WinEdit Object
- WinEditor Object
- WinList Object
- WinListView Object
- WinMenu Object
- WinMenuItem Object
- WinObject Object
- WinRadioGroup Object
- WinRadioButton Object
- WinScrollBar Object
- WinSpin Object
- WinTab Object

- WinTool Bar Object
- WinTreeView Object

Window Object

A standard Window.

Associated Methods:

- Activate Method
- Check Method
- Click Method
- Close Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetROProperty Method
- GetTOProperty Method
- Maximize Method
- Minimize Method
- Move Method
- Output Method
- Resize Method
- Restore Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- hWnd Property
- Object Property

Activate Method

Activates a window or dialog box.

Syntax

Window.Activate

Example

In the following example activates the "Notepad" window.

```
Window("Notepad").Activate
```

Check Method

Executes a checkpoint.

Syntax

Check checkpoint(*name*)

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a check box entitled "Roundtrip Ticket".

```
Check Checkpoint("Roundtrip Ticket")
```

Click Method

Clicks on an object.

Syntax

Window.Click *x*, *y*, [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

Close Method

Closes a window or dialog box.

Syntax

Window.Close

Example

In the following example closes the "Notepad" window.

```
Window("Notepad").Close
```

DblClick Method

Double-clicks on an object.

Syntax

Window.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "Test" window.

```
Window("Test").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

Window.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the window from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags the object from coordinates 10, 20 within the "Test" window and drops the object at coordinates 30, 40 within the "OtherWindow".

```
Window("Test").Drag 10, 20
Window("OtherWindow").Drop 30, 40
```

or within the same window.

```
Window("Test").Drag 10, 20
Window("Test").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

Window.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags the object from coordinates 10, 20 within the “Test” window and drops the object at coordinates 30, 40 within the “OtherWindow”.

```
Window("Test").Drag 10, 20
Window("OtherWindow").Drop 30, 40
```

or within the same window.

```
Window("Test").Drag 10, 20
Window("Test").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

Window.Exist ([*timeout*])

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "Test" window. If the object exists a message box appears confirming its appearance.

```
If Window("Test").Exist
msgbox("The window exists.")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Window.GetROProperty(*Property* **,[***in_PropData***])**

Argument	Type	Description
Property	String	Object property whose value is retrieved.
in_PropData	N/A	Not in use at this time - optional.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the *x* coordinate of the “Test” window.

```
x = Window("Test").GetROProperty ("x")
```

GetTOProperty Method

Returns the value of the specified property for an object. The value is taken from the Object Repository.

Notes:

GetTOProperty differs from the **GetROProperty** method. **GetTOProperty** returns the value from the test object's description. **GetROProperty** returns the current property value of the object in the application during the test run.

Only properties that are included in the test object description can be retrieved. For more information on adding properties to a test object description, refer to the the *QuickTest User's Guide*.

Syntax

Window.GetTOProperty(*Property***)**

Argument	Type	Description
Property	String	Property whose value is retrieved from the object description.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the RegExpWndClass property.

```
Dim ObjectName
RegExpWndClass = Window("Test").GetTOProperty("RegExpWndClass")
```

Maximize Method

Maximizes a window or dialog box to fill the entire screen.

Syntax

Window.Maximize

Example

In the following example maximizes the "Notepad" window.

```
Window("Notepad").Maximize
```

Minimize Method

Minimizes a window or dialog box to an icon.

Syntax

Window.Minimize

Example

In the following example minimizes the "Notepad" window.

```
Window("Notepad").Minimize
```

Move Method

Moves a window or dialog box to the specified absolute location on the screen.

Syntax

Window.Move *x, y*

Argument	Type	Description
x	long	The new horizontal pixel location for the top left corner of the window or dialog box.
y	long	The new vertical pixel location for the top left corner of the window or dialog box.

Example

The following example moves the "Notepad" window.

```
Window("Notepad").Move 659, 35
```

Output Method

Inserts the value of an object into an output value column.

Syntax

Window(*description***).Output Checkpoint(***column _name***)**

Argument	Type	Description
column_name	string	The name of the output value column.

Example

The following example uses the **Output** method to place a text item into an output value column entitled *Submit*.

```
Window("Submit").Output Checkpoint("Submit")
```

Resize Method

Resizes a window or dialog box to the specified dimensions.

Syntax

Window.Resize *width*, *height*

Argument	Type	Description
width	long	The new width of the window, in pixels.
height	long	The new height of the window, in pixels.

Example

The following example resizes the "Notepad" window.

```
Window("Notepad").Resize 296, 348
```

Restore Method

Restores a window or dialog box to its previous size.

Syntax

Window.Restore

Example

In the following example restores the "Notepad" window to its previous size

```
Window("Notepad").Restore
```

SetTOProperty Method

Sets the specified value of a property for an item.

Notes:

SetTOProperty changes the property values used to identify an object during the test run. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Only properties that are included in the test object description can be set. For more information on adding properties to a test object description, refer to the the *QuickTest User's Guide*.

Syntax

Window.SetTOProperty(*Property, Value***)**

Argument	Type	Description
Property	String	Test object property to set.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the index of a window's description.

```
Window("Test").SetTOProperty("Index", 2)
```

Type Method

Sends the keyboard input to the window.

Syntax

Window.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example sends the string, Hello world, to the Notepad window.

```
Window("Notepad").Type "Hello world"
```

The following example uses key strokes to close the Notepad window.

```
Window("Notepad").Type micAltDwn & micF4 & micAltUp
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

Window.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the run-time object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits for the window "Test" to be enabled or for 3 milliseconds to pass, whichever comes first.

```
Window("Test").WaitProperty "enabled", true, 3000
```

hWnd Property

Returns a handle to a run-time object's window

Note: This property is useful when you need to call Win32 API functions

---

Syntax

Window.hWnd

Example

The following example uses the **hWnd** property to change the title of the window

```
'Declare WinAPI SetWindowText function
Extern.Declare micLong, "SetWindowText", "user32.dll", "SetWindowTextA",
micHwnd, micString
'Get handle of the "Notepad" window
handle = Window("Notepad").hWnd
'Call SetWindowText function to change the window title
Extern.SetWindowText handle, "Some other title"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Window(*description***).Object.***Method_to_activate***()**
or

myObj=Window(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your Window. Use the following statements to activate the **Click** method:

```
Dim MyWindow
Set MyWindow=Browser("Mercury Tours").Page("Mercury
Tours").Image("Login").Object
MyWindow.Click
```

Dialog Object

A Windows dialog box.

Associated Methods:

- Activate Method
- Click Method
- Close Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method

- GetROProperty Method
- SetTOProperty Method
- Maximize Method
- Minimize Method
- Move Method
- Resize Method
- Restore Method
- Type Method
- WaitProperty Method

Associated Properties:

- hWnd Property
- Object Property

Activate Method

Activates a window or dialog box.

Syntax

Dialog.Activate

Example

In the following example activates the "Notepad" window.

```
Window("Notepad").Activate
```

Click Method

Clicks on an object.

Syntax

Dialog.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

Close Method

Closes a window or dialog box.

Syntax

Dialog.Close

Example

In the following example closes the "Notepad" window.

```
Window("Notepad").Close
```

DblClick Method

Double-clicks on an object.

Syntax

Dialog.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

Dialog.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

Dialog.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

Dialog.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Dialog.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Dialog.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Dialog.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty ("Name", 0, "Long", "", "1")
```

Maximize Method

Maximizes a window or dialog box to fill the entire screen.

Syntax

Dialog.Maximize

Example

In the following example maximizes the "Notepad" window.

```
Window("Notepad").Maximize
```

Minimize Method

Minimizes a window or dialog box to an icon.

Syntax

Dialog.Minimize
The object is either the standard Window or Dialog object.

Example

In the following example minimizes the "Notepad" window.

```
Window("Notepad").Minimize
```

Move Method

Moves a window or dialog box to the specified absolute location on the screen.

Syntax

Dialog.Move *x, y*

Argument	Type	Description
x	long	The new horizontal pixel location for the top left corner of the window or dialog box.
y	long	The new vertical pixel location for the top left corner of the window or dialog box.

Example

The following example resizes the "Notepad" window.

```
Window("Notepad").Move 659, 35
```

Resize Method

Resizes a window or dialog box to the specified dimensions.

Syntax

Dialog.Resize *width*, *height*

Argument	Type	Description
width	long	The new width of the window, in pixels
height	long	the new height of the window, in pixels

Example

The following example resizes the "Notepad" window.

```
Window("Notepad").Resize 296, 348
```

Restore Method

Restores a window or dialog box to its previous size.

Syntax

Dialog.Restore

Example

In the following example restores the "Notepad" window to its previous size

```
Window("Notepad").Restore
```

Type Method

Types the specified string in the object.

Syntax

Dialog.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example sends the string Hello world to the Properties dialog box.

```
Browser("Untitled").Dialog("Properties").Type "Hello world"
```

The following example uses key strokes to close the "Properties" dialog.

```
Browser("Untitled").Dialog("Properties").Type micAltDwn & micF4 & micAltUp
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

Dialog.WaitProperty (*property* , *value* , *time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

hWnd Property

Returns a handle to a run-time object's window

Note: This property is useful when you need to call Win32 API functions

---

Syntax

Dialog.hWnd

Example

The following example uses the **hWnd** property to change the title of the dialog

```
'Declare WinAPI SetWindowText function
Extern.Declare micLong, "SetWindowText", "user32.dll", "SetWindowTextA",
micHwnd, micString
'Get handle of the "Internet Options" dialog
hDlg = Browser("Untitled").Dialog("Internet Options").hWnd
'Call SetWindowText function to change the dialog title
Extern.SetWindowText hDlg, "Some other title"
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Dialog(*description***).Object.***Method_to_activate***()**
or

myObj=Dialog(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinButton Object

A Windows button object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- hWnd Property
- Object Property

Click Method

Clicks on an object.

Syntax

WinButton.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinButton.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinButton.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinButton.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinButton.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinButton.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" button.

```
Browser("MyPage").Dialog("Settings").WinButton("Advanced").MouseMove 20,
30
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinButton.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty ("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinButton.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty ("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinButton.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example presses the **OK** button by pressing the Enter key

```
Browser("Untitled").Dialog("Properties").WinButton("OK").Type micReturn
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinButton.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

hWnd Property

Returns a handle to a run-time object's window

Note: This property is useful when you need to call Win32 API functions

---

Syntax

WinButton.hWnd

Example

The following example uses the **hWnd** property to disable the "Fonts" button

```
'Declare WinAPI EnableWindow function
Extern.Declare micLong, "EnableWindow", "user32.dll", "EnableWindow",
micHwnd, micLong
'Get handle of the "Fonts" button
hButton = Browser("Untitled").Dialog("Internet
Options").WinButton("Fonts...").hWnd
'Call EnableWindow function with the second parameter 0 to disable the button
Extern.EnableWindow hButton, 0
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinButton(*description***).Object.***Method_to_activate***()**
or

myObj=WinButton(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinCheckBox Object

A Windows check box object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinCheckBox.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinCheckBox.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinCheckBox.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinCheckBox.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinCheckBox.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinCheckBox.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinCheckBox.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" check box.

```
Browser("MyPage").Dialog("Settings").WinCheckBox("Advanced").MouseMove
20, 30
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinCheckBox.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

Set Method

Sets the value of a Windows check box.

Syntax

WinCheckBox.Set " *value* **"**

Argument	Type	Description
value	string	The value to be assigned to the check box. The value can be: "ON" , "OFF" , or "DIMMED" (for three-state check boxes).

Example

The following example sets the "Include subfolders" check box in the "Find: All Files" dialog box to ON.

```
Dialog("Find: All Files").WinCheckBox("Include &subfolders").Set "ON"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinCheckBox.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty ("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinCheckBox.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example selects the **Internet Explorer** check box by pressing the Space key.

```
Browser("Untitled").Dialog("Internet Options").WinCheckBox("Internet
Explorer").Type " "
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinCheckBox.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Syntax

WinCheckBox(*description***).Object.***Method_to_activate***()**
or

myObj=WinCheckBox(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinComboBox Object

A Windows combo box object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetContent Method
- GetTOProperty Method
- GetSelection Method
- ItemsCount Method
- MouseMove Method
- Select Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinComboBox.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinComboBox.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinComboBox.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinComboBox.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinComboBox.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinComboBox.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinComboBox.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinComboBox.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100, Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinComboBox.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinComboBox.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" combo box.

```
Browser("MyPage").Dialog("Settings").WinComboBox("Advanced").MouseMove 20, 30
```

Select Method

Selects an item from the object's list.

Syntax

WinComboBox.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```

or

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinComboBox.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty ("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinComboBox.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty ("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinComboBox.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example uses key strokes to select the **Microsoft Outlook** item from the **Newsgroups** combo box.

```
Browser("Untitled").Dialog("Internet
Options").WinComboBox("Newsgroups:").Type "Microsoft Outlook"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinComboBox.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinComboBox(*description***).Object.***Method_to_activate***()**
or

myObj=WinComboBox(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinContextMenu Object

A Windows context menu object.

Associated Methods:

- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method

- GetROProperty Method
- SetTOProperty Method

Associated Properties:

- Object Property

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinContextMenu.Drag x, y [, button]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinContextMenu.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinContextMenu.Exist([*timeout*]**)**

The object can be any object from any environment.

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinContextMenu.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinContextMenu.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinContextMenu.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinContextMenu(*description***).Object.***Method_to_activate***()**
or

myObj=WinContextMenu(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinEdit Object

A Windows edit object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method

- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- GetROProperty Method
- SetTOProperty Method
- Set Method
- SetSecure Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinEdit.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinEdit.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinEdit.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinEdit.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinEdit.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinEdit.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinEdit.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" edit box.

```
Browser("MyPage").Dialog("Settings").WinEdit("Advanced").MouseMove 20, 30
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinEdit.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinEdit.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Set Method

Sets the value of an edit field

Syntax

WinEdit.Set " *text* **"**

Argument	Type	Description
text	string	The text to be entered in the edit field.

Example

The following example sets the file name "document.txt" in the "File name:" edit box.

```
Dialog("Open").WinEdit("File &name:").Set "document.txt"
```

SetSecure Method

Sets the encrypted value of an edit field.

Note: The SetSecure method is recorded when a password or other secure text is entered. The text is encrypted while recording, and decrypted during the test run.

Syntax

WinEdit.SetSecure " *SecureText* **"**

Argument	Type	Description
SecureText	string	The encrypted text to be entered in the edit field.

Example

The following example sets a password in "Enter password" dialog box.

```
Dialog("Enter password").WinEdit("Password:").SetSecure
"cvrt55ccrf5656edsd"
```

Type Method

Types the specified string in the object.

Syntax

WinEdit.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see “Type Constants,” on page 497.

Example

The following example sends the string Hello world to the **Select a Web** edit box.

```
Browser("Untitled").Dialog("Internet Options").WinEdit("Select a Web").Type
"Hello world"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinEdit.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Syntax

WinEdit(*description***).Object.***Method_to_activate***()**
or

myObj=WinEdit(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinEditor Object

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- MouseMove Method
- SetCaretPos Method
- SetSelection Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinEditor.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinEditor.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinEditor.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinEditor.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinEditor.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" editor.

```
Browser("MyPage").Dialog("Settings").WinEditor("Advanced").MouseMove 20, 30
```

SetCaretPos Method

Places the cursor at the specified point in an edit object.

Syntax

WinEditor.SetCaretPos *Line, Col*

Argument	Type	Description
Line	Number	The zero-based row position at which the insertion point is placed.
Col	Number	The zero-based column position at which the insertion point is placed.

Example

The following example types a sentence at the beginning of the third line of the “RICHEDIT” WinEditor object.

```
Window("WordPad").WinEditor("RICHEDIT").SetCaretPos 3, 0
Window("WordPad").WinEditor("RICHEDIT").Type "This text will be inserted at the beginning of the line 3"
```

SetSelection Method

Selects text in an edit object.

Syntax

WinEditor.SetSelection *StartLine, StartCol, EndLine, EndCol*

Argument	Type	Description
StartLine	Number	The row at which the selection starts.
StartCol	Number	The column at which the selection starts.
EndLine	Number	The row at which the selection ends.
EndCol	Number	The column at which the selection ends.

Example

The following example selects text from the beginning of line 3 through column 16 or line four, and then presses the Delete key to delete the selected text.

```
Window("WordPad").WinEditor("RICHEDIT").SetSelection 3, 0, 4, 16
Window("WordPad").WinEditor("RICHEDIT").Type micDel
```

Type Method

Types the specified string in the object.

Syntax

WinEditor.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example sends the string Hello world to the **Message body** editor.

```
Browser("Untitled").Dialog("New Mail").WinEditor("Message body").Type "Hello world"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinEditor.WaitProperty(*property , value , time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinEditor(*description***).Object.***Method_to_activate***()**
or

myObj=WinEditor(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinList Object

A Windows list object.

Associated Methods:

- Activate Method
- Click Method
- DblClick Method
- Deselect Method
- Drag Method
- Drop Method
- Exist Method
- ExtendSelect Method
- GetContent Method
- GetTOProperty Method

- GetSelection Method
- ItemsCount Method
- MouseMove Method
- Select Method
- SelectRange Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Activate Method

Activates (double-clicks) an item in the object's list.

Syntax

WinList.Activate *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example double-clicks on the "Myfile item in the "Files" list.

```
Dialog("Open").WinList("Files").Activate "MyFile"
```
or

```
Dialog("Open").WinList("Files").Activate 2
```

Click Method

Clicks on an object.

Syntax

WinList.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinList.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Deselect Method

Deselects an item in the object's list.

Syntax

WinList.Deselect *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example deselects a selected item from the "Files" List.

```
Dialog("Open").WinList("Files").Deselect "MyFile"
```
or

```
Dialog("Open").WinList("Files").Deselect 3
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinList.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinList.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinList.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

ExtendSelect Method

Selects an additional item from a multi-selection list.

Syntax

WinList.ExtendSelect *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example adds an item to a multiple-selection list in the "Files" List.

```
Dialog("Open").WinList("Files").ExtendSelect "MyFile"
```
or

```
Dialog("Open").WinList("Files").ExtendSelect 2
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinList.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinList.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinList.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100, Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinList.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinList.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" list.

```
Browser("MyPage").Dialog("Settings").WinList("Advanced").MouseMove 20, 30
```

Select Method

Selects an item from the object's list.

Syntax

WinList.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"

or

Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7

SelectRange Method

Selects all list items between (and including) the two specified items.

Note: This method is applicable only to lists that allow multiple selection.

Syntax

WinList.SelectRange [*item1 ,*] *item2*

Argument	Type	Description
item1	string	**optional.** The first item of the range. If the item1 argument is not specified, then the currently selected item is the first item of the range. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.
item2	string	The last item of the range. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects all the items between (and including) "Red" and "Blue".

```
Dialog("Select Color").WinList("Colors").SelectRange "Red", "Blue"
```

The following example selects the items between the currently selected item and item number 7.

```
Dialog("Select Color").WinList("Colors").SelectRange 7
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinList.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinList.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinList.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example uses keystrokes to select the **Microsoft Outlook** item from the **Newsgroups** list.

```
Browser("Untitled").Dialog("Internet Options").WinList("Newsgroups:").Type
"Microsoft Outlook"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinList.WaitProperty(*property* , *value* , *time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinList(*description***).Object.***Method_to_activate***()**
or

myObj=WinList(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinListView Object

A Windows ListView object.

Associated Methods:

- Activate Method
- Click Method
- DblClick Method
- Deselect Method
- Drag Method

- Drop Method
- Exist Method
- ExtendSelect Method
- GetCheckMarks Method
- GetContent Method
- GetTOProperty Method
- GetSelection Method
- ItemsCount Method
- Select Method
- SelectRange Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Activate Method

Activates (double-clicks) an item in the object's list.

Syntax

WinListView.Activate *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example double-clicks on the "Myfile item in the "Files" list.

```
Dialog("Open").WinList("Files").Activate "MyFile"
```
or

```
Dialog("Open").WinList("Files").Activate 2
```

Click Method

Clicks on an object.

Syntax

WinListView.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinListView.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Deselect Method

Deselects an item in the object's list.

Syntax

WinListView.Deselect *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example deselects a selected item from the "Files" List.

```
Dialog("Open").WinList("Files").Deselect "MyFile"
```
or
```
Dialog("Open").WinList("Files").Deselect 3
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinListView.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinListView.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinListView.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

ExtendSelect Method

Selects an additional item from a multi-selection list.

Syntax

WinListView.ExtendSelect *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example adds an item to a multiple-selection list in the "Files" List.

Dialog("Open").WinList("Files").ExtendSelect "MyFile"

or

Dialog("Open").WinList("Files").ExtendSelect 2

GetCheckMarks Method

Retrieves the number and the value of items marked as checked.

Syntax

WinListView.GetCheckMarks

Example

In the following example, the string "checked" gets values of checked items from "SysListView".

```
string 100,checked
checked = Dialog("Lists").WinTreeView("SysListView").GetCheckMarks
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinListView.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinListView.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinListView.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100, Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinListView.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

Select Method

Selects an item from the object's list.

Syntax

WinListView.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```

or

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7
```

SelectRange Method

Selects all list items between (and including) the two specified items.

Note: This method is applicable only to lists that allow multiple selection.

Syntax

WinListView.SelectRange [*item1* ,] *item2*

Argument	Type	Description
item1	string	**optional.** The first item of the range. If the item1 argument is not specified, then the currently selected item is the first item of the range. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.
item2	string	The last item of the range. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects all the items between (and including) "Red" and "Blue".

```
Dialog("Select Color").WinList("Colors").SelectRange "Red", "Blue"
```

The following example selects the items between the currently selected item and item number 7.

```
Dialog("Select Color").WinList("Colors").SelectRange 7
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinListView.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinListView.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinListView.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example uses keystrokes to select the Microsoft Outlook item from the **Newsgroups** list.

```
Browser("Untitled").Dialog("Internet Options").WinListView("Newsgroups:").Type
"Microsoft Outlook"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinListView.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinListView(*description***).Object.***Method_to_activate***()**

or

myObj=WinListView(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinMenu Object

A Windows menu object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- GetROProperty Method
- SetTOProperty Method
- WaitProperty Method

Associated Properties:

➤ Object Property

Click Method

Clicks on an object.

Syntax

WinMenu.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinMenu.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinMenu.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinMenu.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinMenu.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinMenu.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinMenu.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinMenu.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinMenu.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Syntax

WinMenu(*description***).Object.***Method_to_activate***()**

or

myObj=WinMenu(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinMenuItem Object

A Windows menu item object.

Associated Methods:

- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- GetROProperty Method
- Select Method
- SetTOProperty Method

Associated Properties:

- Object Property

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinMenuItem.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinMenuItem.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinMenuItem.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinMenuItem.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinMenuItem.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

Select Method

Selects an item from the object's list.

Syntax

WinMenuItem.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```
or

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinMenuItem.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinMenuItem(*description***).Object.***Method_to_activate***()**

or

myObj=WinMenuItem(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinObject Object

A standard (Windows) object.

Associated Methods:

- Check Method
- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetROProperty Method
- GetTOProperty Method
- Output Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Methods:

- hWnd Property
- Object Property

Check Method

Executes a checkpoint.

Syntax

Check checkpoint(*name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a check box entitled "Roundtrip Ticket".

```
Check Checkpoint("Roundtrip Ticket")
```

Click Method

Clicks on an object.

Syntax

WinObject.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinObject.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinObject.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinObject.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinObject.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinObject.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinObject.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinObject.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" object.

```
Browser("MyPage").Dialog("Settings").WinObject("Advanced").MouseMove 20, 30
```

Output Method

Inserts the value of an object into an output value column.

Syntax

WinObject.Output Checkpoint(*column _name***)**

Argument	Type	Description
column_name	string	The name of the output value column.

Example

The following example uses the **Output** method to place a text item into an output value column entitled *Submit*.

```
Browser("QA Home Page").Page("QA Home Page").WebButton("Submit").Output Checkpoint ("Submit")
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinObject.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinObject.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see “Type Constants,” on page 497.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinObject.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

hWnd Property

Returns a handle to a run-time object's window

Note: This property is useful when you need to call Win32 API functions

---

Syntax

WinObject.hWnd

Example

The following example uses the **hWnd** property to disable the "Flight" object

```
'Declare WinAPI EnableWindow function
Extern.Declare micLong, "EnableWindow", "user32.dll", "EnableWindow",
micHwnd, micLong
'Get handle of the "Flight" object
hObject = Browser("Untitled").WinObject("Flight").hWnd
'Call EnableWindow function with the second parameter 0 to disable the button
Extern.EnableWindow hObject, 0
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinObject(*description***).Object.***Method_to_activate***()**
or

myObj=Image(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinRadioGroup Object

A Windows radio group object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetContent Method
- GetTOProperty Method
- GetSelection Method
- ItemsCount Method
- MouseMove Method
- Select Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinRadioGroup.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinRadioGroup.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinRadioGroup.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinRadioGroup.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinRadioGroup.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinRadioGroup.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinRadioGroup.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinRadioGroup.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100,Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinRadioGroup.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinRadioGroup.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20,30) inside the "Advanced" radio button.

```
Browser("MyPage").Dialog("Settings").WinRadioGroup("Advanced").MouseMov
e 20, 30
```

Select Method

Selects an item from the object's list.

Syntax

WinRadioGroup.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```
or

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinRadioGroup.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinRadioGroup.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinRadioGroup.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinRadioGroup.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinRadioGroup(*description***).Object.***Method_to_activate***()**

or

myObj=WinRadioGroup(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinRadioButton Object

A Windows radio button object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- GetROProperty Method
- Set Method

- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinRadioButton.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinRadioButton.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinRadioButton.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinRadioButton.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinRadioButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinRadioButton.GetTOProperty(*Property,* [**,** *Flags***,** *Type***,** *ParameterName*])

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinRadioButton.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" radio button.

```
Browser("MyPage").Dialog("Settings").WinRadioButton("Advanced").MouseMov
e 20, 30
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinRadioButton.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

Set Method

Sets the radio button (as the selected radio button).

Syntax

WinRadioButton.Set

Example

The following example sets the "Not Shared" radio button.

```
Window("Window").Dialog("System Properties").WinRadioBut("Not Shared").Set
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinRadioButton.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinRadioButton.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example selects the **Internet Explorer** radio button by pressing the Space key.

```
Browser("Untitled").Dialog("Internet Options").WinRadioButton("Internet
Explorer").Type " "
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinRadioButton.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinRadioButton(*description***).Object.***Method_to_activate***()**

or

myObj=WinRadioButton(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinScrollBar Object

A Windows scroll bar object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- NextLine Method
- NextPage Method

- PrevLine Method
- PrevPage Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinScrollBar.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinScrollBar.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinScrollBar.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinScrollBar.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinScrollBar.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinScrollBar.GetTOProperty(*Property,* [**,** *Flags***,** *Type***,** *ParameterName*])

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinScrollBar.MouseMove *X, Y*

Argument	Type	Description
X, Y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" scroll bar.

```
Browser("MyPage").Dialog("Settings").WinScrollBar("Advanced").MouseMove 20, 30
```

NextLine Method

Moves the scroll bar downward, or to the right, the specified number of lines.

Syntax

WinScrollBar.NextLine *NumOfLines*

Argument	Type	Description
NumOfLines	long	The number of lines to move the scroll bar.

Example

The following example moves the scroll bar 7 lines down.

```
Dialog("scroll").WinScrollBar("ScrollBar").NextLine 7
```

NextPage Method

Moves the scroll bar downward, or to the right, the specified number of pages.

Syntax

WinScrollBar.NextPage *NumOfPages*

Argument	Type	Description
NumOfPages	long	The number of pages to move the scroll bar.

Example

The following example moves the scroll bar 7 pages down.

```
Dialog("scroll").WinScrollBar("ScrollBar").NextPage 7
```

PrevLine Method

Moves the scroll bar upward, or to the left, the specified number of lines.

Syntax

WinScrollBar.PrevLine *NumOfLines*

Argument	Type	Description
NumOfLines	long	The number of lines to move the scroll bar.

Example

The following example moves the scroll bar 5 lines up.

```
Dialog("scroll").WinScrollBar("ScrollBar").PrevLine 5
```

PrevPage Method

Moves the scroll bar upward, or to the left, the specified number of pages.

Syntax

WinScrollBar.PrevPage *NumOfPages*

Argument	Type	Description
NumOfPages	long	The number of pages to move the scroll bar.

Example

The following example moves the scroll bar 5 pages up.

```
Dialog("scroll").WinScrollBar("ScrollBar").PrevPage 5
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinScrollBar.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

Set Method

Sets the scroll bar position.

Syntax

WinScrollBar.Set " *value* **"**

Argument	Type	Description
value	long	The scroll bar position to be set.

Example

The following example sets the scroll bar to position 77.

```
Dialog("scroll").WinScrollBar("ScrollBar").Set 77
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinScrollBar.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinScrollBar.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinScrollBar.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to “The Object Property,” on page 671.

Syntax

WinScrollBar(*description***).Object.***Method_to_activate***()**

or

myObj=WinScrollBar(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinSpin Object

A Windows Spin object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetTOProperty Method
- MouseMove Method
- Next Method
- Prev Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinSpin.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinSpin.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinSpin.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinSpin.Drop *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinSpin.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinSpin.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinSpin.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" spin object.

```
Browser("MyPage").Dialog("Settings").WinSpin("Advanced").MouseMove 20, 30
```

Next Method

Sets a Spin object to its next value.

Syntax

WinSpin.Next

Example

In the following example sets the "Spin" object to its next value.

```
Dialog("SpinDialog").WinSpin("Spin").Next
```

Prev Method

Sets a Spin object to its previous value.

Syntax

WinSpin.Prev

Example

In the following example sets the "Spin" object to its previous value.

```
Dialog("SpinDialog").WinSpin("Spin").Prev
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinSpin.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

Set Method

Sets a Spin object to the specified item.

Syntax

WinSpin.Set *value*

Argument	Type	Description
value	integer	The value to which the Spin object should be set.

Example

The following example sets the Spin value to 7.

```
Dialog("SpinDialog").WinSpin("Spin"). Set 7
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinSpin.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinSpin.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example decrements the spin value by pressing the down arrow twice.

```
Browser("Untitled").Dialog("Internet
Options").Dialog("Settings").WinSpin("msctls_updown32").Type micDwn &
micDwn
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinSpin.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinSpin(*description***).Object.***Method_to_activate***()**

or

myObj=WinSpin(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinTab Object

A Windows tab object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method
- GetContent Method
- GetTOProperty Method
- GetSelection Method
- ItemsCount Method

- MouseMove Method
- Select Method
- GetROProperty Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinTab.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinTab.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinTab.Drag *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinTab.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinTab.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinTab.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinTab.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinTab.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100, Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinTab.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinTab.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" object.

```
Browser("MyPage").Dialog("Settings").WinTab("Advanced").MouseMove 20, 30
```

Select Method

Selects an item from the object's list.

Syntax

WinTab.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```

or

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 22
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinTab.GetROProperty(*Property, in_PropData*)

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinTab.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinTab.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example moves two tabs to the left by pressing the left arrow button twice.

```
Browser("Untitled").Dialog("Internet Options").WinTab("SysTabControl32").Type
micLeft & micLeft
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinTab.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinTab(*description***).Object.***Method_to_activate***()**
or

myObj=WinTab(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinTool Bar Object

A Windows toolbar object.

Associated Methods:

- Click Method
- DblClick Method
- Drag Method
- Drop Method
- Exist Method

- GetTOProperty Method
- GetROProperty Method
- MouseMove Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Click Method

Clicks on an object.

Syntax

WinToolBar.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

DblClick Method

Double-clicks on an object.

Syntax

WinToolBar.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinToolBar.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinToolBar.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinToolBar.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinToolBar.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinToolBar.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinToolBar.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" toolbar.

```
Browser("MyPage").Dialog("Settings").WinToolBar("Advanced").MouseMove 20, 30
```

Press Method

Presses a toolbar button.

Syntax

WinToolBar.Press *ToolBarButton* [, *MouseButton*]

Argument	Type	Description
ToolBarButton	Variant	The toolbar button to press. The logical name (with quotes) or numeric index (without quotes) can denote the button. The first item in a list is numbered 0. The logical name reflects the button's attached text (tooltip).
MouseButton	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example presses the Bold toolbar button on the Formatting toolbar.

```
Window("WordPad").WinToolbar("Formatting").Press "Bold"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinToolBar.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinToolBar.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinToolBar.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example, the program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinToolBar(*description***).Object.***Method_to_activate***()**
or

myObj=WinToolBar(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

WinTreeView Object

A Windows TreeView object.

Associated Methods:

- Activate Method
- Click Method
- Collapse Method
- DblClick Method
- Drag Method

- Drop Method
- Exist Method
- Expand Method
- GetCheckMarks Method
- GetContent Method
- GetROProperty Method
- GetTOProperty Method
- GetSelection Method
- ItemsCount Method
- MouseMove Method
- Select Method
- SetItemState Method
- SetTOProperty Method
- Type Method
- WaitProperty Method

Associated Properties:

- Object Property

Activate Method

Activates (double-clicks) an item in the object's list.

Syntax

WinTreeView.Activate *item*

Argument	Type	Description
Item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example double-clicks on the "Myfile item in the "Files" list.

```
Dialog("Open").WinList("Files").Activate "MyFile"
```
or

```
Dialog("Open").WinList("Files").Activate 2
```

Click Method

Clicks on an object.

Syntax

WinTreeView.Click *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the click.
y	Number	The y-coordinate of the click.
button	Integer	**Optional.** The button used to click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example clicks a right mouse button at coordinates 47, 131 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").Click 47, 131, 1
```

Collapse Method

Hides sub-items, such as individual files in a folder, in an expanded list.

Syntax

WinTreeView(*description***).Collapse** *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example collapses the "Ins" item.

```
Window("Exploring").WinTreeView("SysTreeView32").Collapse "Ins"
```

DblClick Method

Double-clicks on an object.

Syntax

WinTreeView.DblClick *x*, *y* [,*button*]

Argument	Type	Description
x	Number	The x-coordinate of the double-click.
y	Number	The y-coordinate of the double-click.
button	Integer	**Optional.** The button used to double-click the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example double-clicks a right mouse button at coordinates 73, 120 on "SysListView32" object.

```
Window("Exploring").WinListView("SysListView32").DblClick 73, 120, 1
```

Drag Method

Performs the "drag" part of a drag and drop operation.

Syntax

WinTreeView.Drag *x, y* [, *button*]

Argument	Type	Description
x, y	Number	The coordinates within the object from which the object is dragged
button	Integer	**Optional.** The button used to drag the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

Drop Method

Performs the "drop" part of a drag and drop operation.

Syntax

WinTreeView.Drop *x, y* [*, button*]

Argument	Type	Description
x, y	Number	The coordinates of the object onto which the object is dropped
button	Integer	**Optional.** The button that is released to drop the object. 0=LEFT, 1=RIGHT, 2=MIDDLE. **Default=0 (LEFT)**

Example

The following example drags and drops "SomeObject".

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeObject").Drop 30, 40
```

or

```
Dialog("MyDialog").WinObject("SomeObject").Drag 10, 20
Dialog("MyDialog").WinObject("SomeOtherObject").Drop 30, 40
```

Exist Method

Checks that an object exists.

Syntax

WinTreeView.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

Expand Method

Displays hidden sub-items, such as individual files in a folder, in a list.

Syntax

WinTreeView.Expand *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example expands the "Ins" item.

```
Window("Exploring").WinTreeView("SysTreeView32").Expand "Ins"
```

GetCheckMarks Method

Retrieves the number and the value of items marked as checked.

Syntax

WinTreeView.GetCheckMarks

Example

In the following example, the string "checked" gets values of checked items from "SysListView".

```
string 100, checked
checked = Dialog("Lists").WinTreeView("SysListView").GetCheckMarks
```

GetContent Method

Returns all of the items in the object's list.

Syntax

WinTreeView.GetContent

Example

The following example returns the items in the "Files" list.

```
Contents=Dialog("Open").WinList("Files").GetContent
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WinTreeView.GetROProperty(*Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the state of a check box object.

```
CheckState = Browser.Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty("Value")
' CheckState contains "OFF"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WinTreeView.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser("Mercury Interactive").Page("Mercury Interactive")
.Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetSelection Method

Returns all of the selected items in the object's list.

Syntax

WinTreeView.GetSelection

Example

The following example returns the selected items in the "Files" list.

```
Dialog("Open").WinList("Files").Select "bin"
Dialog("Open").WinList("Files").ExtendSelect "dat"
Dialog("Open").WinList("Files").ExtendSelect "encrypt"
string 100, Selection
Selection = Dialog("Open").WinList("Files").GetSelection
```

ItemsCount Method

Returns the number of items in the object's list.

Syntax

WinTreeView.ItemsCount

Example

The following example returns the number of items in the "Files" list.

```
NumberOfItems = Dialog("Open").WinList("Files").ItemsCount
```

MouseMove Method

Moves the mouse pointer to the designated position inside the object.

Syntax

WinTreeView.MouseMove *x, y*

Argument	Type	Description
x, y	number	The position of the mouse pointer, expressed as x and y (pixel) coordinates. Note that the specified coordinates are relative to the upper left corner of the object.

Example

The following example moves the mouse pointer to the position (20, 30) inside the "Advanced" tree view.

```
Browser("MyPage").Dialog("Settings").WinTreeView("Advanced").MouseMove 20, 30
```

Select Method

Selects an item from the object's list.

Syntax

WinTreeView.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the item "22" from the "Size" combo box.

```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select "22"
```
or
```
Window("Notepad").Dialog("Font").WinCombo("&Size:").Select 7
```

SetItemState Method

Sets the state of a check box icon of the specified item in a treeview.

Syntax

WinTreeView.SetItemState *Item, State*

Argument	Type	Description
Item	variant	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.
State	number	1 (checked) or 0 (unchecked)

Example

The following example clears the check box icon of the Child item in the treeview.

```
Dialog("Search").WinTreeView("SysTreeView32").SetItemState "Root1;Child", 0
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WinTreeView.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

Type Method

Types the specified string in the object.

Syntax

WinTreeView.Type *string*

Argument	Type	Description
string	string	The text string and/or constants representing non-alphanumeric keys. For a list of constants, see "Type Constants," on page 497.

Example

The following example expands all tree nodes by pressing the * key.

```
Browser("Untitled").Dialog("Internet Options").WinTreeView("Settings").Type "*"
```

WaitProperty Method

Waits for an object property to attain the specified value and then continues the test run. If the specified value is not attained, the method waits until the specified time expires before continuing the test.

Syntax

WinTreeView.WaitProperty(*property, value, time* **)**

Argument	Type	Description
property	String	Any property of the object.
value	String	The property value for which the method waits.
time	Number	The interval, in seconds, before the next statement is executed.

Example

In the following example program waits until the "No" button is enabled or 3 milliseconds pass.

```
Dialog("Notepad").WinButton("&No").WaitProperty "enabled", ON, 3
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WinTreeView(*description***).Object.***Method_to_activate***()**
or

myObj=WinTreeView(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statement to activate the **Click** method:

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Object.Click
```

Referenced Information

Type Constants

Constant	Action
micCtrlDwn	presses the Ctrl key
micCtrlUp	releases the Ctrl key
micLCtrlDwn	presses the left Ctrl key
micLCtrlUp	releases the left Ctrl key
micRCtrlDwn	presses the right Ctrl key
micRCtrlUp	releases the right Ctrl key
micAltDwn	presses the Alt key
micAltUp	releases the Alt key
micLAltDwn	presses the left Alt key
micLAltUp	releases the left Alt key
micRAltDwn	presses the right Alt key
micRAltUp	releases the right Alt key
micShiftDwn	presses the Shift key
micShiftUp	releases the Shift key
micLShiftDwn	presses the left Shift key
micLShiftUp	releases the left Shift key
micRShiftDwn	presses the right Shift key
micRShiftUp	releases the right Shift key
micIns	presses the Insert key
micDel	presses the Delete key
micHome	presses the Home key

Constant	Action
micEnd	presses the End key
micPgUp	presses the Page Up key
micPgDwn	presses the Page Down key
micUp	presses the up arrow key
micDwn	presses the down arrow key
micLeft	presses the left arrow key
micRight	presses the right arrow key
micEsc	presses the Esc key
micBack	presses the Backspace key
micReturn	presses the Return key
micTab	presses the Tab key
micBreak	presses the Break key
micPause	presses the Pause key
micPrintScr	presses the Print Screen key
micF1	presses the F1 key
micF2	presses the F2 key
micF3	presses the F3 key
micF4	presses the F4 key
micF5	presses the F5 key
micF6	presses the F5 key
micF7	presses the F6 key
micF8	presses the F7 key
micF9	presses the F8 key
micF10	presses the F9 key

Constant	Action
micF11	presses the F10 key
micF12	presses the F11 key
micNumLockOn	turns on the Num Lock
micCapsLockOn	turns on the Caps Lock
micScrollOn	turns on the Scroll Lock
micNumLockOff	turns off the Num Lock
micCapsLockOff	turns off the Caps Lock
micScrollOff	turns off the Scroll Lock

7

Utility

The following objects can be used to set testing and result reporting preferences during the test.

- Crypt Object
- Extern Object
- JavaUtil Object
- OptionalStep Object
- Reporter Object
- Setting Object
- WebPackage (Setting) Object
- TSLTest Object
- VirtualButton Object
- VirtualCheckBox Object
- VirtualList Object
- VirtualObject Object
- VirtualRadioButton Object
- VirtualTable Object
- WebUtil Object

The following utility functions help you control your test.

- DescribeResult Function
- ExitAction Function

- ExitActionIteration Function
- ExitGlobalIteration Function
- ExitRun Function
- GetLastError Function
- InvokeApplication Function
- RunAction Function
- SetLastError Function
- Wait Function

Crypt Object

The object used to encrypt strings.

Associated Methods:

- Encrypt Method

Encrypt Method

Encrypts a string.

Syntax

Crypt.Encrypt(*data***)**

Argument	Type	Description
Data	String	The encrypted string.

Return Value

String

Example

In the following example, a password is taken from a database and encrypted using the **Encrypt** method, and then placed in the password edit box using the SetSecure method.

```
pwd = "GetPasswordfromSomewhere"
e_pwd = Crypt.Encrypt(pwd)
Browser("dfgd").Dialog("pass").WinEdit("pwd").SetSecure e_pwd
```

Extern Object

Calls external procedures from an external dynamic-link library (DLL).

Associated Methods:

➤ Declare Method

Any Declared method (see Declare method for more information)

Declare Method

Declares references to external procedures in a dynamic-link library (DLL).

Once you use the Declare method for a method, you can use the Extern object to call the declared method.

Syntax

Extern.Declare(*RetType, MethodName, LibName, Alias* [[, *ArgType(s)*], ...] **)**

Argument	Type	Description
RetType	String	Data type of the value returned by the method.
MethodName	String	Any valid procedure name.
LibName	String	Name of the DLL or code resource that contains the declared procedure.
Alias	String	Name of the procedure in the DLL or code resource. **Note:** DLL entry points are case sensitive. **Note:** If *Alias* is an empty string, *MethodName* is used as the Alias.
ArgType(s)	String	A list of data types representing the data types of the arguments that are passed to the procedure when it is called. For available argument types, see "Declare Argument Types," on page 556.

Example

The following example uses the **Extern.Declare** and **Extern.<declared method>** methods to change the title of the Notepad window.

```
'Declare FindWindow method
Extern.Declare micHwnd, "FindWindow", "user32.dll", "FindWindowA",
micString, micString
'Declare SetWindowText method
Extern.Declare micLong, "SetWindowText", "user32.dll", "SetWindowTextA",
micHwnd, micString
'Get HWND of the Notepad window
hwnd = Extern.FindWindow("Notepad", vbNullString)
if hwnd = 0 then
MsgBox "Notepad window not found"
end if
'Change the title of the notepad window
res = Extern.SetWindowText(hwnd, "kuku")
```

The following example checks when the cursor is an hour glass.

```
extern.Declare
micLong,"GetForegroundWindow","user32.dll","GetForegroundWindow"
extern.Declare
micLong,"AttachThreadInput","user32.dll","AttachThreadInput",micLong,micLon
g,micLong
extern.Declare
micLong,"GetWindowThreadProcessId","user32.dll","GetWindowThreadProces
sId",micLong,micLong
extern.Declare
micLong,"GetCurrentThreadId","kernel32.dll","GetCurrentThreadId"
extern.Declare micLong,"GetCursor","user32.dll","GetCursor"

function get_cursor() hwnd = extern.GetForegroundWindow()
        pid = extern.GetWindowThreadProcessId(hWnd, NULL)
        thread_id=extern.GetCurrentThreadId()
        extern.AttachThreadInput pid,thread_id,TRUE
        get_cursor=extern.GetCursor()
        extern.AttachThreadInput pid,thread_id,FALSE
end function
Msgbox get_cursor()
```

JavaUtil Object

A Java Utility object used for setting or retrieving Java testing preference settings.

Associated Methods:

- GetAUTVar Method
- SetAUTVar Method

GetAUTVar Method

Retrieves the value of the specified variable.

Note: You can view additional Java options in the Java tab of the Options dialog box (**Tools** > **Options**) and in the Advanced Java Options dialog box (**Tools** > **Options** > **Java** > **Advanced**).

Syntax

JavaUtil.**GetAUTVar(** *variable* **)**

Argument	Type	Description
variable	String	The name of the variable to be retrieved. For a listing of the variables and the possible values for each variable, see "Java Test Variables," on page 558.

Example

The following example uses the **GetAUTVar** method to retrieve the setting for the "column_number" Settings variable.

```
JavaUtil.SetAUTVar "column_number", "2"
MsgBox JavaUtil.GetAUTVar("column_number")
```

SetAUTVar Method

Sets the value of the specified variable.

Note: You can view additional Java options in the Java tab of the Options dialog box (**Tools** > **Options**) and in the Advanced Java Options dialog box (**Tools** > **Options** > **Java** > **Advanced**).

Syntax

JavaUtil.**SetAUTVar** *variable, value*

Argument	Type	Description
variable	String	The name of the test setting to be set. For a listing of the variables see "Java Test Variables," on page 558.
value	Variant	The value to be assigned to the specified variable. For a listing of possible values for each variable, see "Java Test Variables," on page 558.

Example

The following example uses the **SetAUTVar** method to set the value for the "column_number" variable.

```
JavaUtil.SetAUTVar "column_number", "2"
MsgBox JavaUtil.GetAUTVar("column_number")
```

OptionalStep Object

The OptionalStep object causes a step to be bypassed if an object in that step is not found.

Syntax

OptionalStep.*Object...*

Example

The following example uses the **OptionalStep** object to make the *Paris* selection from the *Depart* WebList an optional step.

```
OptionalStep.Browser("Mercury Tours").Page("Find
Flights").WebList("depart").Select "Paris"
```

Reporter Object

The object used for sending information to the test report in theRecorder.

Associated Methods:

- ReportEvent Method

ReportEvent Method

Reports status to the test report.

Syntax

Reporter.ReportEvent *EventStatus, ReportStepName, Details*

Argument	Type	Description
EventStatus	Number	Status of the report step: **0 - Passed:** Causes the status of this step to be passed and sends the specified message to the report. **1 - Failed:** The status of the step is "Failed": Causes the status of this step to be failed and sends the specified message to the report. When this step runs, the test fails. **2 - General:** Sends a message to the report. without affecting the pass/fail status of the test. **3 - Warning:** Sends a warning message to the report, but does not cause the test to stop running, and does not affect the pass/fail status of the test.
ReportStepName	String	Name of the intended step in the report (object name).
Details	String	Description of the report event. The string will be displayed in the step details frame in the test report.

Example

The following example uses the ReportEvent method to report a failed step to the test report.

```
Reporter.ReportEvent 1, "Custom Step", "The user-defined step failed"
```

Setting Object

Enables you to modify test settings during the test run. The settings are a dictionary of keys and their values, to be used as global variables.

Associated Methods:

- Add Method
- Exists Method
- Remove Method

Associated Properties:

- AutomaticLinkRun Property
- DefaultLoadTime Property
- DefaultTimeOut Property
- Item Property
- ReplayType Property
- WebTimeOut Property

Add Method

Adds and sets a user-defined run-time setting.

Syntax

Setting.Add *SettingName, SettingValue*

Argument	Type	Description
SettingName	String	Assigns a name to the new run-time setting.
SettingValue	String	Assigns a value to the new run-time setting.

Example

The following example uses the **Add** method to create a new setting called "TesterName", assigns the value "JohnSmith" to the setting, and writes the value of the setting to the report.

```
Setting.Add "TesterName","JohnSmith"
Tester = Setting("TesterName")
Reporter.ReportEvent 1, "The tester is", Tester.
```

Exists Method

Determines if a key exists in the dictionary.

Syntax

Setting.Exists(*Key***)**

Argument	Type	Description
Key	String	The name of the key to check.

Example

The following example checks to see if the IterNumber key exists. If it doesn't exist, it adds it to the dictionary and assigns the value 1. If it already exists, it increments the value.

```
If Not Setting.Exists("IterNum") then
    Setting.Add("IterNum"), 1 'create "IterNum" and assign to 1
else
    Setting.Item("IterNum")=Setting.Item("IterNum")+1 'increment "IterNum"
end if
```

Remove Method

Removes the value of a user-defined run-time setting.

Syntax

Setting.Remove *SettingName*

Argument	Type	Description
SettingName	String	The name of the run-time setting.

Example

The following example uses the **Remove** method to remove the "TesterName" setting.

```
Setting.Remove "TesterName"
```

AutomaticLinkRun Property

Determines whether automatically created checkpoints will be executed during the test run. AutomaticLinkRun is a global testing setting.

Syntax

To retrieve the setting:

Setting = **Setting**("**AutomaticLinkRun**")

To set the setting:

Setting("**AutomaticLinkRun**") = *Setting*

Argument	Type	Possible Values	Description
Setting	Variant	1, TRUE 0, FALSE	Automatic checkpoints are executed. Automatic checkpoints are not executed.

Example

The following example shows the use of the AutomaticLinkRun property to enable automatic page checks.

```
Setting("AutomaticLinkRun") = 1
```

DefaultLoadTime Property

Sets or retrieves the setting for the add to load time option. DefaultLoadTime is a global testing setting. Changes to this setting will be maintained for the remainder of the current testing session.

Syntax

To retrieve the setting:

Setting("**DefaultLoadTime**")

To set the setting:

Setting("**DefaultLoadTime**") = seconds

Part	Type	Description
seconds	Number	Indicates the time to add, in milliseconds, to the load time during recording. This option is a safeguard that prevents the test from failing in the event that the amount of time it takes for a page to load during the run session is higher than the amount of time it took during the record session.

Example

The following example sets the default load time to 3 seconds.

```
Setting("DefaultLoadTime") = 3
```

DefaultTimeOut Property

Sets or retrieves the delay for finding objects. DefaultTimeOut is a per-test setting.

Syntax

To retrieve the setting:

Setting = **Setting**("**DefaultTimeout**")

To set the setting:

Setting("**DefaultTimeout**") = *milliseconds*

Argument	Type	Description
milliseconds	Number	maximum time in milliseconds to wait before it is determined that an object cannot be found.

Example

The following example sets the default timeout to four seconds.

```
Setting("DefaultTimeout") = 4000
```

Item Property

The key/value pair from the dictionary.

Syntax

Setting.Item(*Key***)**

Argument	Type	Description
Key	String	The name of the object's key.

Example

The following example checks to see if the IterNumber key exists. If it doesn't exist, it adds it to the dictionary and assigns the value 1. If it already exists, it uses the item property of the setting object, and increments the iteration value.

```
With Setting'accessing the "Setting" Object
    If Not .exist("IterNumber") then
   .Add("IterNumber"),1
    Else
   .Item("IterNumber")=.Item("IterNumber")+1
    End if
End With
```

ReplayType Property

Indicates the run mode used during the test run.

Syntax

Setting.WebPackage("ReplayType") = *modesetting*

Argument	Type	Possible Values	Description
modesetting	Number	1 2	**Default.** Sets the test to run by events. Sets the test to run by mouse operation

Example

The following example shows the use of the Replay Type property in order to send the coordinates of the image button of a form during the test run. After the image click, the run mode is returned to the default.

```
Setting.WebPackage("ReplayType") = 2
Browser("Method of Payment").Page("Flight Confirmation").Image("Book
Another").Click 131, 14
Setting.WebPackage("ReplayType") = 1
```

WebTimeOut Property

Sets or retrieves the delay for navigating to a URL address. WebTimeOut is a per-test setting.

Syntax

To retrieve the setting:

Setting = **Setting**("**WebTimeout**")

To set the setting:

Setting("**WebTimeout**") = *milliseconds*

Argument	Type	Description
milliseconds	Number	maximum time in seconds to wait before it is determined that a URL address cannot be found.

Example

The following example sets the Web timeout to five seconds.

```
Setting("WebTimeout") = 5000
```

WebPackage (Setting) Object

Sets the run mode for testing Web sites to events (default) or mouse operation.

Associated Properties:

➤ ReplayType Property

ReplayType Property

Indicates the run mode used during the test run.

Syntax

Setting.WebPackage("ReplayType") = *modesetting*

Argument	Type	Possible Values	Description
modesetting	Constant	1 2	**Default**. Sets the test to run by events. Sets the test to run by mouse operation

Example

The following example shows the use of the Replay Type property in order to send the coordinates of the image button of a form during the test run. After the image click, the run mode is returned to the default.

```
Setting.WebPackage("ReplayType") = 2
Browser("Method of Payment").Page("Flight Confirmation").Image("Book
Another").Click 131, 14
Setting.WebPackage("ReplayType") = 1
```

TSLTest Object

The TSLTest object enables you to run WinRunner tests and call WinRunner functions from QuickTest.

Associated Methods:

- CallFunc Method
- RunTest Method

CallFunc Method

Enables you to call a TSL function in a WinRunner compiled module from QuickTest.

Syntax

TSLTest.**CallFunc(** *ModulePath, TSLFunction* [*, Parameters*] **)**

Argument	Type	Description
ModulePath	String	The path of the WinRunner compiled module.
TSLFunction	String	The function in the compiled module to call.
Parameters	String	1-8 WinRunner test parameters.

Example

In the following example, QuickTest calls the "MessageBoxA" function, defined in the Windows API module that resides in the *lib\win32api* folder in the WR installation directory. This function creates a message box whose caption is its third parameter and whose text string is its second parameter.

```
TSLTest.CallFunc ("C:\Program Files\Mercury
Interactive\WinRunner\lib\win32api", "MessageBoxA", "0", "The function
MessageBoxA from user32 works correctly.", "MessageBoxA function", "3")
```

RunTest Method

Enables you to run a WinRunner test from QuickTest.

Syntax

TSLTest.RunTest(*TestPath* , *TestSet* [, *Parameters*] **)**

Argument	Type	Description
TestPath	String	The path of the WinRunner test.
TestSet	String	The test set within TestDirector, in which test runs are stored. Note that this argument is relevant only when working with a test in a TestDirector project. When the test is not saved in TestDirector, this parameter is ignored.
Parameters	String	1-8 WinRunner function argument.

Example

In the following example, QuickTest runs the “test1” WinRunner test.

```
TSLTest.RunTest "D\:test1", ""
```

VirtualButton Object

An object mapped to the virtual button class.

Associated Methods

- Click Method
- DblClick Method
- Exist Method
- GetTOProperty Method
- SetTOProperty Method

Click Method

Clicks on a virtual object.

Syntax

VirtualButton.Click

Example

The following example clicks the "button" VirtualButton.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualButton("button").Click
```

DblClick Method

Double-clicks on an object.

Syntax

VirtualButton.DblClick

Example

The following example double-clicks the "button" VirtualButton.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualButton("button").DblClick
```

Exist Method

Checks that an object exists.

Syntax

VirtualButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualButton.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualButton.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

VirtualCheckBox Object

An object mapped to the virtual check box class.

Associated Methods

- Exist Method
- GetTOProperty Method
- Set Method
- SetTOProperty Method

Exist Method

Checks that an object exists.

Syntax

VirtualCheckBox.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualCheckBox.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
Parameter Name	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Set Method

Toggles the check box state.

Syntax

VirtualCheckBox.Set " *value* **"**

Argument	Type	Description
value	string	always “TOGGLE”.

Example

The following example toggles the “checkbox” VirtualCheckBox (from ON to OFF or the reverse).

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualCheckBox("check
box").Set"TOGGLE"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualCheckBox.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

VirtualList Object

An object mapped to the virtual list class.

Associated Methods

- Activate Method
- Exist Method
- GetTOProperty Method
- Select Method
- SetTOProperty Method

Activate Method

Activates (double-clicks) an item in the object's list.

Syntax

VirtualList.Activate *item*

Argument	Type	Description
item	string or long	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example double-clicks on the 4th item in the list.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualList("list").Activate 3
```

Exist Method

Checks that an object exists.

Syntax

VirtualList.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualList.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Select Method

Selects an item from the object's list.

Syntax

VirtualList.Select *item*

Argument	Type	Description
item	string	The item to select from the list. The logical name (with quotes) or numeric index (without quotes) can denote the item. The first item in a list is numbered 0.

Example

The following example selects the 4th item in the list.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualList("list").Select 3
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualList.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

VirtualObject Object

An object mapped to the virtual object class.

Associated Methods

- Click Method
- DblClick Method
- Exist Method
- GetTOProperty Method
- SetTOProperty Method

Click Method

Clicks on the specified object.

Syntax

VirtualObject.Click

Example

The following example clicks the “button” VirtualButton.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualObject("MyObject").Click
```

DblClick Method

Double-clicks on the specified object.

Syntax

VirtualButton.DblClick

Example

The following example double-clicks the “button” VirtualButton.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualObject("MyObject").DblClick
```

Exist Method

Checks that an object exists.

Syntax

VirtualObject.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualObject.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualObject.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

VirtualRadioButton Object

An object mapped to the virtual radiobutton class.

Associated Methods

- Exist Method
- GetTOProperty Method
- Set Method
- SetTOProperty Method

Exist Method

Checks that an object exists.

Syntax

VirtualRadioButton.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualRadioButton.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Set Method

Sets the radio button (as the selected radio button).

Syntax

VirtualRadioButton.Set

Example

The following example sets the "Not Shared" radio button.

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualRadioButton("radio button").Set
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualRadioButton.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

VirtualTable Object

An object mapped to the virtual table class.

Associated Methods

- ActivateCell Method
- Exist Method
- GetTOProperty Method
- SetTOProperty Method
- SelectCell Method

ActivateCell Method

Activates the specified cell in the table.

Syntax

VirtualTable.ActivateCell *row* , *col*

Argument	Type	Description
row	variant	Indicates the row number or the row header label
col	variant	Indicates the column number or the column header label

Example

The following example activates the cell in row 2, column 2 of the VirtualTable:

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualTable("table").ActivateCell  2, 2
```

Exist Method

Checks that an object exists.

Syntax

VirtualTable.Exist([*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the Exist method to determine the existence of the "username" Java edit box. If the object exists a message box appears confirming its appearance.

```
If
Browser("Browser").Page("Page").Applet("login.html").JavaEdit("username").Exi
st Then
msgbox("The object exists.")
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

VirtualTable.GetTOProperty(*Property,* **[,** *Flags***,** *Type***,** *ParameterName***])**

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Flags	Number	**Optional.** Flags indicating the method of retrieval.
Type	String	**Optional.** Variable type to return (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional.** Parameter name in the Data Table to which to output the value.

Return Value

Variant

Example

The following example uses the GetTOProperty method to retrieve the HTML-name of the image.

```
Dim ObjectName
ObjectName = Browser.Page("Mercury
Interactive").Image("MercuryLogo").GetTOProperty("Name")
' ObjectName contains "Enterprise Application Testing Solutions"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

VirtualTable.SetTOProperty *Property, Value,* [**,** *Flags***,** *Type***,** *ParameterName*]

Argument	Type	Description
Property	String	Property to retrieve from the object description.
Value	Variant	The value to assign to the listed property.
Flags	Number	**Optional input parameter.** Flags indicating the set method.
Type	String	**Optional input parameter.** Variable type to set (i.e. "String", "Boolean", etc.).
ParameterName	String	**Optional input parameter.** Parameter name in the Data-Table to which to write the value.

Example

The following example uses the SetTOProperty method to set the index of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", 0, "Long", "", "1")
```

SelectCell Method

Selects the specified cell in the table.

Syntax

VirtualTable.SelectCell *row* , *col*

Argument	Type	Description
row	variant	Indicates the row number or the row header label
col	variant	Indicates the column number or the column header label

Example

The following example selects the cell in row 2, column 2 of the VirtualTable:

```
Browser("Demo of Vo Object").Page("Demo of Vo
Object").ActiveX("VoDemoFormX").VirtualTable("table").SelectCell  2, 2
```

WebUtil Object

The WebUtil object enables you to set or modify Web related settings.

Associated Methods:

- AddCookie Method
- DeleteCookie Method
- DeleteCookies Method
- GetCookies Method
- SetAuthenticationPassword Method
- SetAuthenticationUserName Method

AddCookie Method

Adds a cookie to the test.

Syntax

WebUtil.**AddCookie** *domain, cookie_info*

Argument	Type	Description
domain	String	The name of the domain that the host uses.
cookie_info	String	The cookie information in the format: NAME=VALUE; expires=DATE;path=PATH; domain=DOMAIN_NAME; secure

Example

In the following example, a cookie is inserted into the test.

```
WebUtil.AddCookie "www.dummy.com", "MyCookie=12345678; expires=Wdy,
22-Dec-2001 00:00:00 GMT; domain=.dummy.com; path=/ ;secure"
```

In the following example a cookie is erased from the test by inserting an expired date into the method.

```
WebUtil.AddCookie "www.dummy.com", "MyCookie=12345678; expires=Wdy,
22-Dec-1999 00:00:00 GMT; domain=.dummy.com; path=/ ;secure"
```

DeleteCookie Method

Deletes a cookie from the cookie table.

Syntax

WebUtil.DeleteCookie *domain, cookie_info*

Argument	Type	Description
domain	String	Name of the domain that the host uses.
cookie_info	String	The cookie information in the format: NAME=VALUE; expires=DATE;path=PATH; domain=DOMAIN_NAME; secure

Example

The following example deletes the second cookie from the table.

```
WebUtil.DeleteCookie "www.dummy.com", "SecondCookie"
```

DeleteCookies Method

Deletes all cookies for the browser being used in the test run.

Syntax

WebUtil.DeleteCookies

Example

The following statement deletes all of the cookies from the cookie table.

```
WebUtil.DeleteCookies
```

GetCookies Method

Retrieves the list of cookies for the browser being used in the test run.

Syntax

WebUtil.**GetCookie** *domain*

Argument	Type	Description
domain	String	Name of the domain that the host uses.

Example

The following example retrieves the cookie list and prints a message.

```
Dim CookiesList
CookiesList = WebUtil.GetCookies "www.dummy.com"
MsgBox("CookiesList contains" & CookiesList)
```

SetAuthenticationPassword Method

Allows the user to enter a password to be used throughout the test, or until it is reset.

Syntax

WebUtil.**SetAuthenticationPassword** *password*

Argument	Type	Description
password	String	Password for authentication.

Example

The following example sets the password to “abracadabra.”

```
WebUtil.SetAuthenticationPassword("abracadabra")
```

SetAuthenticationUserName Method

Allows the user to enter a user name to be used throughout the test, or until it is reset.

Syntax

WebUtil.SetAuthenticationUserName *username*

Argument	Type	Description
username	String	User name for authentication.

Example

The following example sets the user name to “merlin.”

```
WebUtil.SetAuthenticationUserName("merlin")
```

Utility Functions

- DescribeResult Function
- ExitAction Function
- ExitActionIteration Function
- ExitGlobalIteration Function
- ExitRun Function
- GetLastError Function
- InvokeApplication Function
- RunAction Function
- SetLastError Function
- Wait Function

DescribeResult Function

Returns a text description of the last error.

Syntax

DescribeResult(*Error* **)**

Argument	Type	Description
Error	Integer	The error.

Return Value

String

Example

In the following example, the script fails to run successfully because the image "Login" was not found on the Mercury Tours page. Using the DescribeResult function, a description of the error is placed into a message box.

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Click 19, 55
x = GetLastError
msgbox(DescribeResult(x))
```

ExitAction Function

Exits the current action, regardless of its iteration attributes.

Note: The **ExitAction** statement and its return value are displayed in the Test Results.

Syntax

ExitAction(*RetVal* **)**

Argument	Type	Description
RetVal	string	The action's return value.

Example

In the following example, the CheckForm action calls the GetFormVersion action. The GetFormVersion action checks whether the form is in the new version or the old version. If IsNewForm is TRUE, then the action uses the **ExitAction** function to return the value 2. If IsNewForm is FALSE, then the action uses the **ExitAction** function to return the value 1. The CheckForm action stores the value returned by the ExitAction statement in the FormVersion variable, and uses it in the following steps.

```
' Action "CheckForm"
FormVersion = RunAction("GetFormVersion", oneIteration)
If FormVersion = 2 Then
    Call RunAction("CheckOldForm", oneIteration)
Else
    Call RunAction("CheckNewForm", oneIteration)
End If
```

```
' Action "GetFormVersion"
If IsNewForm Then
    ExitAction(2)
Else
    ExitAction(1)
End If
```

ExitActionIteration Function

Exits the current iteration of the action.

Note: The **ExitActionIteration** statement and its return value are displayed in the Test Results.

Syntax

ExitActionIteration(*RetVal* **)**

Argument	Type	Description
RetVal	string	The action's return value.

Example

The following example uses the **ExitActionIteration** function to stop running the current action iteration when the value in the current row of the City column in the action's data sheet is Paris.

```
CurrentCity = DataTable("City", dtLocalSheet )
If CurrentCity = "Paris" Then ExitActionIteration("Skipping Paris")
```

ExitGlobalIteration Function

Exits the current global iteration.

Note: The **ExitGlobalIteration** statement and its return value are displayed in the Test Results.

Syntax

ExitGlobalIteration(*RetVal* **)**

Argument	Type	Description
RetVal	string	The return value.

Example

The following example uses the **ExitGlobalIteration** function to stop running the current iteration of the test when the value in the current row of the City column in the global data table is Paris.

```
CurrentCity = DataTable("City")
If CurrentCity = "Paris" Then ExitGlobalIteration("Skipping Paris")
```

ExitRun Function

Exits the test, regardless of its iteration attributes.

Note: The **ExitRun** statement and its return value are displayed in the Test Results.

Syntax

ExitRun(*RetVal* **)**

Argument	Type	Description
RetVal	string	The return value.

Example

The current example, exits the test and returns the value 0.

```
ExitRun(0)
```

GetLastError Function

Returns the most recent error.

Syntax

GetLastError

Return Value

Number

Example

In the following example, the script fails to run successfully because the "Login" image was not found on the Mercury Tours page. Using the **GetLastError** function, the error number is inserted into a message box.

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Logi").Click 19, 55
x = GetLastError
msgbox(x)
```

InvokeApplication Function

Invokes an executable application.

Syntax

InvokeApplication(*Application* [,*Path*] **)**

Argument	Type	Description
Application	String	The name of the executable file
Path	String	The path to the application's directory.

Return Value

Boolean. If the function fails to open the application, False is returned.

Example

The following example uses the **InvokeApplication** function to open a text file entitled paste.txt in the Notepad application.

```
InvokeApplication "notepad paste.txt", "\\harpo\bavly\tmp"
```

RunAction Function

Runs the specified action in the test.

Syntax

RunAction(*ActionName, Iteration_Mode* **[,** *Iteration_Range***])**

Argument	Type	Description
ActionName	string	The name of the action.
Iteration_Mode	pre-defined constant	oneIteration (0) - runs only one iteration using the first row in the data table. rngIterations (1) - runs iterations according to the third parameter **Note:** *Iteration_Mode* is required when calling an external action, but optional when calling a local action (from within the test).
Iteration_Range	variant	**Optional.** The row range (i.e. 1-7), or rngAll (1--1). (valid only when the *Iteration_Mode* is rngIterations (1)).

Example

The following example calls the SearchFlight action, and runs one iteration of the action.

```
RunAction "SearchFlight", oneIteration
```

SetLastError Function

Sets an error into the test script.

Syntax

SetLastError(*Error* **)**

Argument	Type	Description
Error	Integer	The error.

Example

The following example uses the **SetLastError** function to place the "Out of memory" error into the test script. For a list of VBScript errors, refer to the Microsoft VBScript Reference in the Help menu.

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Click 11, -18
SetLastError(7)
Browser("Mercury Tours").Page("Welcome to Mercury").Image("Search
Flights").Click
```

Wait Function

Initiates a pause during the run of a test.

Syntax

Wait seconds [**,** *milliseconds*]

Argument	Type	Description
seconds	long	Number of seconds to wait between steps.
milliseconds	long	Number of milliseconds to wait between steps.

Example

The following example uses the **Wait** method to pause 10 seconds between selecting a departure city and an arrival city during the test run.

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select
"London"
Wait(10)
Browser("Mercury Tours").Page("Find Flights").WebList("arrive").Select "New
York"
```

Referenced Information

Declare Argument Types

The following argument types can be used for the Extern.Declare ArgType(s) argument.

Type	Description
micVoid	void (RetType only)
micInteger	int
micLong	long
micFloat	float
micDouble	double
micString	CHAR*
micDispatch	IDispatch*
micWideString	WChar*
micChar	char
micUnknown	IUnknown
micHwnd	HWND
micVPtr	void*
micShort	short

Type	Description
micWord	WORD
micDWord	DWORD
micByte	BYTE
micWParam	WPARAM
micLParam	LPARAM
micLResult	LRESULT
micUChar	unsigned char
micULong	unsigned long
micUShort	unsigned short
micUInteger	unsigned int

Java Test Variables

The following variables can be set or retrieved using the **JavaUtil.SetAUTVar** or **JavaUtil.GetAUTVar** methods.

Variable Name	Description	Possible Values
column_number	Minimum number of columns for a table to be considered a table object. Otherwise the edit fields are treated as separate objects. (Oracle only)	Any integer **Default = 2**
edit_replay_mode	Controls how actions are performed on edit fields.	"S" - sends the setValue () method to set a value of the edit object. "P" - sends KeyPressed event to the object for every character from the input string. "T" - sends KeyTyped events to the object for every character from the input string. "R" - sends KeyReleased event to the object for every character from the input string. "F" - generates a FocusLost event at the end of method execution.
max_column_gap	The maximum number of pixels between objects in a table to be considered a column. (Oracle only)	Any integer **Default = 12**
max_line_deviation	The maximum number of pixels between objects to be considered to be on a single line. (Oracle only)	Any integer **Default = 8**

Variable Name	Description	Possible Values
max_list_columns	The maximum number of columns in an Oracle LOV object to be considered a list. A larger number constitutes a table. (Oracle only)	Any integer **Default = 99**
max_row_gap	The maximum number of pixels between objects to be considered one table row. (Oracle only)	Any integer **Default = 12**
max_text_distance	Sets the maximum distance in pixels, to look for attached text.	Any integer **Default = 100**

8

Web

The following objects can be used when testing Web objects.

- Browser Object
- Page Object
- Frame Object
- Image Object
- Link Object
- WebArea Object
- WebButton Object
- WebCheckBox Object
- WebEdit Object
- WebElement Object
- WebFile Object
- WebList Object
- WebRadioGroup Object
- WebTable Object

Browser Object

The title of the first Web page.

Associated Methods:

- Back Method
- Check Method
- Close Method
- Exist Method
- Forward Method
- FullScreen Method
- GetTOProperty Method
- Home Method
- Navigate Method
- GetROProperty Method
- Refresh Method
- SetTOProperty Method
- Sync Method

Associated Properties:

- Object Property

Back Method

Navigates to the last accessed page in the browser. Identical to pressing the browser's "Back" button.

Syntax

Browser(*description***).Back**
The object is always the **Browser** object.

Example

The following example uses the **Back** method.

```
Browser("Mercury Tours").Back
```

Check Method

Executes a checkpoint.

Syntax

Check checkpoint(*name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a check box entitled "Roundtrip Ticket".

```
Check Checkpoint("Roundtrip Ticket")
```

Close Method

Closes the browser window.

Syntax

Browser(*description***).Close**

Example

The following example uses the **Close** method to close the Mercury Tours application.

```
Browser("Mercury Tours").Page("Search Results").Image("reserveFlights").Click
41, 20
Browser("Mercury Tours").Page("Method of Payment").Image("buyFlights").Click
11, 5
Browser("Mercury Tours").Close
```

Exist Method

Checks that an object exists.

Syntax

Browser(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "Mercury Tours" browser. If the object exists, a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Exist Then
  MsgBox "The browser exists."
End If
```

Forward Method

Navigates to the next page in the browser history list. Identical to pressing the browser's "Forward" button.

Syntax

Browser(*description***).Forward**.
The object is always the **Browser** object.

Example

The following example uses the **Forward** method to navigate to the following page after the **Back** method has been used.

```
Browser("Mercury Tours").Page("Find Flights").Image("findFlights").Click 103, 13
Browser("Mercury Tours").Back
Browser("Mercury Tours").Page("Find
Flights").WebRadioGroup("seatType").Select "Coach"
Browser("Mercury Tours").Forward
Browser("Mercury Tours").Page("Search Results").Check CheckPoint("Search
Results")
```

FullScreen Method

Displays the browser in full-screen mode.

Syntax

Browser(*description***).FullScreen**

Example

The following example uses the **FullScreen** method.

```
Browser("Web Testing").FullScreen
```

GetTOProperty Method

Returns the specified value of a property for a test object. The value is taken from the Object Repository.

Syntax

Browser(*description***).GetTOProperty(***Property* **)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the browser's name property from the Object Repository.

```
BrowserName = Browser("Mercury Tours").GetTOProperty("Name")
' BrowserName contains "Mercury Tours"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Home Method

Navigates to the homepage configured in the browser's settings. Identical to selecting the browser's 'Home' button.

Syntax

Browser(*description***).Home**

Example

The following illustrates the use of the **Home** method.

```
Browser("Web Testing").Home
```

Navigate Method

Opens a specified URL in the browser.

Syntax

Browser(*description***).Navigate(***URL* **[,** *FrameName***], [***headers***], [***post data***])**

Argument	Type	Description
URL	String	URL to open in the browser.
FrameName	String	Frame in the browser to which the URL will be sent. The frame name is specified in the NAME argument of the <FRAME> tag.
headers	String	Header data, such as cookies and content-type.
post data	String	Data that accompanies a Post request.

Example

The following example uses the **Navigate** method to navigate to the Mercury Interactive Web site.

```
Browser("Mercury Tours").Navigate("www.merc-int.com")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Browser(*description***).GetROProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the silent property of a browser object.

```
IsSilent = Browser("Mercury Tours").GetROProperty("silent")
```

Refresh Method

Refreshes the objects in the browser.

Syntax

Browser(*description***).Refresh**

Example

The following example uses the **Refresh** method to refresh the objects in the "Mercury Tours" browser.

```
Browser("Mercury Tours").Refresh
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Browser(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	string	Property to add to the object description.
Value	variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of the browser's description in run-time.

```
Browser("Mercury Tours").SetTOProperty("Name", "my_browser")
```

Sync Method

Waits for the browser to complete the current navigation.

Syntax

Browser(*description***).Sync**

Example

The following example uses the **Sync** method to wait for the Find Flights page to synchronize before navigating to the next page.

```
Browser("Mercury Tours").Page("Find Flights").Image("findFlights").Click 88, 13
Browser("Mercury Tours").Page("Find Flights").Sync
Browser("Mercury Tours").Page("Search Results").Check CheckPoint("Search
Results")
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Browser(*description***).Object.***Method_to_activate***()**
or

myObj=Browser(*description***).Object**
myObj.*Method_to_activate***()**

Example

In the following example, suppose the **Refresh** method is supported for your browser. Use the following statements to activate the **Refresh** method:

```
Dim MyBrowser
Set MyBrowser=Browser("Mercury Tours").Object
MyBrowser.Refresh
```

Page Object

An HTML page.

Associated Methods:

- Check Method
- Exist Method
- GetTOProperty Method
- GetROProperty Method

- SetTOProperty Method
- Sync Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

Page(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a page checkpoint.

```
Browser("Mercury Tours").Page("Find Flights").Check Checkpoint("Roundtrip Ticket")
```

Exist Method

Checks that an object exists.

Syntax

Page(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "Find Flights" page. If the object exists a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Find Flights").Exist Then
  MsgBox "The page exists."
End If
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Page(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the page's name property from the Object Repository.

```
PageName = Browser("Mercury Tours").Page("Mercury
Tours").GetTOProperty("Name")
' PageName contains "Mercury Tours"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Page(*description***).GetROProperty(***Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the URL of a Web page.

```
PageURL = Browser("Mercury Tours").Page("Mercury Tours").GetROProperty
("URL")
'PageURL contains "http://mercurytours.mercuryinteractive.com"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Page(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the number of images in a page.

```
Browser("Mercury Tours").Page("Mercury Tours").SetTOProperty("number of
images", 5)
```

Sync Method

Waits for the browser to complete the current navigation.

Syntax

Page(*description***).Sync**

Example

The following example uses the **Sync** method to wait for the Mercury Tours page to synchronize before navigating to the next page.

```
Browser("Mercury Tours").Page("Mercury Tours").Sync
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to"The Object Property," on page 671.

Syntax

Page(*description***).Object.***Method_to_activate***()**
or

myObj=Page(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the Page object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/obj_document.asp
```

Example

In the following example, suppose the **Close** method is supported for your page. Use the following statements to activate the **Close** method:

```
Dim MyPage
Set MyPage=Browser("Mercury Tours").Page("Mercury Tours").Object
MyPage.Close
```

Frame Object

The frame.

Associated Methods:

- Check Method
- Exist Method
- GetTOProperty Method
- Output Method
- GetROProperty Method
- SetTOProperty Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

Frame(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a frame called "navbar".

```
Browser("Mercury Tours").Page("Mercury Tours").Frame("navbar").Check
CheckPoint("navbar")
```

Exist Method

Checks that an object exists.

Syntax

Frame(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "menu deutsch" frame. If the frame exists a message box appears confirming its appearance.

```
If Browser("Tibhar-HomePage").Page("Tibhar-HomePage").Frame("menu
deutsch").Exist Then
  MsgBox "The frame exists."
End If
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Frame(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to add to the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the frame's name property from the Object Repository.

```
FrameName = Browser("Tibhar-HomePage").Page("Tibhar-
HomePage").Frame("menu deutsch").GetTOProperty("Name")
' FrameName contains "menu deutsch"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

Output Method

Saves the value of an object to an output value in the data table.

Syntax

Frame(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output value.

Example

The following example uses the **Output** method to place a frame checkpoint into an output value entitled *info*.

```
Browser("Mercury Tours").Page("Mercury Tours").Frame("info").Output
Checkpoint("info")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Frame(*description***).GetROProperty(***Property***, *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the name of a frame's description in run-time.

```
Browser("Tibhar-HomePage").Page("Tibhar-HomePage").Frame("menu
deutsch").GetROProperty ("name")
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Frame(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the source of a frame in run-time.

```
Browser("Tibhar-HomePage").Page("Tibhar-HomePage").Frame("menu
deutsch").SetTOProperty("src", "http:\\tibhar")
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Frame(*description***).Object.***Method_to_activate***()**
or

myObj=Frame(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the **Frame** object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/FRAME.asp

Example

In the following example, suppose the **Clear** method is supported for your frame. Use the following statements to activate the **Clear** method:

```
Dim MyFrame
Set MyFrame=Browser("Tibhar-HomePage").Page("Tibhar-
HomePage").Frame("menu deutsch").Object
MyFrame.clear
```

Image Object

An image with or without a target URL.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method

- GetTOProperty Method
- MouseOver Method
- Output Method
- GetROProperty Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

Image(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for an image entitled "Login".

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Check
Checkpoint ("Login")
```

Click Method

Clicks an object.

Syntax

Image(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "Login" image after a name and password are inserted in the sign up page of the "Mercury Tours" flight application.

```
Browser("MercuryTours").Page("MercuryTours").WebEdit("username").Set"merc
ury"
Browser("Mercury Tours").Page("Mercury
Tours").WebEdit("password").SetSecure "389057c1827c4bc250a95"
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Click 25, 100
```

Exist Method

Checks that an object exists.

Syntax

Image(*description***).Exist(**[*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "Login" image. If the image exists, a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Mercury Tours").Image("Login").Exist Then
  MsgBox "The image exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

Image(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate moving the mouse over an image.

```
Browser("Mercury Tours").Page("Mercury Tours").Image("Login").FireEvent
"onmouseover"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Image(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the image's name property from the Object Repository.

```
ImageName = Browser("Mercury Tours").Page("Mercury
Tours").Image("Login").GetTOProperty("Name")
' ImageName contains "Login"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

Image(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a search button.

```
Browser("Mercury Tours").Page("Find Flights").Image("Search
Flights").MouseOver
```

Output Method

Saves the value of an object to an output value in the data table.

Syntax

Image(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output value column.

Example

The following example uses the **Output** method to place a checkpoint image into an output value column entitled *Search Flights*.

```
Browser("Mercury Tours").Page("Find Flights").Image("Search Flights").Output
Checkpoint ("Search Flights")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Image(*description***).GetROProperty(***Property***, *in_PropData*)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the href property of an image object.

```
Value = Browser("Mercury Tours").Page("Find Flights").Image("Search
Flights").GetROProperty("href")
' Value contains
"http://mercurytours.mercuryinteractive.com/reservations.pperl?page=welcome"'
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Image(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of an image's description in run-time.

```
Browser("Mercury
Tours").Page("FindFlights").WebTable("banner_animated").WebTable("Sun").Im
age("Sun").SetTOProperty("Name", "Sunshine")
```

Submit Method

Submits a form.

Syntax

Image(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from an image.

```
Browser("Web Testing").Page("Mercury Tours").Image("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Image(*description***).Object.***Method_to_activate***()**
or

myObj=Image(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the Image object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/refere
nce/objects/IMG.asp
```

Example

In the following example, suppose the **Click** method is supported for your image. Use the following statements to activate the **Click** method:

```
Dim MyImage
Set MyImage=Browser("Mercury Tours").Page("Mercury
Tours").Image("Login").Object
MyImage.Click
```

Link Object

A hypertext link.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- MouseOver Method
- Output Method
- GetROProperty Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

Link(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a link for a checkpoint entitled "Jobs".

```
Browser("Mercury Interactive").Page("Mercury Interactive").Link("Jobs").Check Checkpoint("Jobs")
```

Click Method

Clicks an object.

Syntax

Link(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "Jobs" link.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").Click 25, 100
```

Exist Method

Checks that an object exists.

Syntax

Link(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "Jobs" link. If the link exists, a message box appears confirming its appearance.

```
If Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").Exist Then
  MsgBox "The link exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

Link(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate clicking a link.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").FireEvent "onclick"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

Link(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the link's name property from the Object Repository.

```
LinkName=Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").GetTOProperty("name")
'LinkName contains "Jobs"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

Link(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a link.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").MouseOver
```

Output Method

Inserts the value of an object into an output parameter.

Syntax

Link(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output parameter.

Example

The following example uses the **Output** method to place a link text item into an output parameter entitled *Jobs*.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").Output Checkpoint("Jobs")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

Link(*description***).GetROProperty(***Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the href of a link object.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").GetROProperty("href")
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

Link(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	string	Property to add to the object description.
Value	variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of a link's description in run-time.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").SetTOProperty("Name", "new jobs")
```

Submit Method

Submits a form.

Syntax

Link(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a hypertext link.

```
Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Link(*description***).Object.***Method_to_activate***()**
or

myObj=Link(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the Link object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/refere
nce/objects/A.asp
```

Example

In the following example, suppose the **href** property is supported for your link. Use the following statements to set the **href** property:

```
Dim MyLink
Set MyLink=Browser("Mercury Technologies").Page("Mercury
Technologies").Link("Jobs").Object
MyLink.href = "http://www.mercury.com"
```

WebArea Object

A section of a picture on a page.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- MouseOver Method
- GetROProperty Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebArea(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a text area called "comments".

```
Browser("Mercury Tours").Page("Mercury Tours").WebArea("comments").Check
Checkpoint("comments")
```

Click Method

Clicks an object.

Syntax

WebArea(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "comments" text area of the "Mercury Tours" flight application.

```
Browser("Mercury Tours").Page("Mercury Tours").WebArea("comments").Click
```

Exist Method

Checks that an object exists.

Syntax

WebArea(*description***).Exist(**[*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "comments" text area. If the object exists a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Mercury Tours").WebArea("comments").Exist
Then
  MsgBox "The comment area exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebArea(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the “Event List,” on page 670.

Example

The following example uses the **FireEvent** method to simulate focusing on a text area.

```
Browser("Mercury Tours").Page("Mercury
Tours").WebArea("comments").FireEvent "onfocus"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebArea(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the Web area's name property from the Object Repository.

```
Browser("Mercury Tours").Page("Mercury
Tours").WebArea("comments").GetTOProperty("Name")
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebArea(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a text area.

```
Browser("Mercury Tours").Page("Mercury
Tours").WebArea("comments").MouseOver
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebArea(*description***).GetROProperty(***Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the state of a text area.

```
val = Browser("Mercury Tours").Page("Mercury
Tours").WebArea("comments").GetROProperty("Value")
' val contains the text in the entry field of the text area "comments"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebArea(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of a text area's description in run-time.

```
Browser("Mercury Tours").Page("Mercury
Tours").WebArea("comments").SetTOProperty("Name", "my comment")
```

Submit Method

Submits a form.

Syntax

WebArea(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a WebArea object.

```
Browser("Web Testing").Page("Mercury Tours").WebArea("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebArea(*description***).Object.***Method_to_activate***()**
or

myObj=WebArea(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebArea object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/AREA.asp

Example

In the following example, suppose the **defaultValue** method is supported for a text area. Use the following statements to activate the **defaultValue** method:

```
Dim MyWebArea, dVal
Set MyWebArea = Browser("Mercury").Page("Mercury").WebArea("Exe
WebArea").Object
Set dVal = MyWebArea.defaultValue
'dVal contains the default value of Exe WebArea
```

WebButton Object

A button.

Associated Methods:

- Check Method
- Click Method
- Exist Method

- FireEvent Method
- GetTOProperty Method
- MouseOver Method
- Output Method
- GetROProperty Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebButton(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a button entitled "Submit".

```
Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").Check Checkpoint("Submit")
```

Click Method

Clicks an object.

Syntax

WebButton(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "Submit" button.

```
Browser("QA Home Page").Page("QA Home Page").WebButton("Submit").Click
```

Exist Method

Checks that an object exists.

Syntax

WebButton(*description***).Exist(**[*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "submit" button. If the object exists a message box appears confirming its appearance.

```
If Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").Exist Then
  MsgBox "The button exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebButton(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the “Event List,” on page 670.

Example

The following example uses the **FireEvent** method to simulate clicking the mouse on a button.

```
Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").FireEvent "onclick"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebButton(*description***).GetTOProperty (***Property* **)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the button's name property from the Object Repository.

```
ButtonName=Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").GetTOProperty("Name")
' ButtonName contains "Submit"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebButton(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a button.

```
Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").MouseOver
```

Output Method

Inserts the value of an object into an output value.

Syntax

WebButton(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output value.

Example

The following example uses the **Output** method to place a text item into an output parameter entitled *Submit*.

```
Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").Output Checkpoint("Submit")
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebButton(*description***).GetROProperty(***Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the value of a WebButton object.

```
ButtonType = Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").GetROProperty("Value")
' ButtonType contains "Submit"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebButton(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of a button's description in run-time.

```
Browser("QA Home Page").Page("QA Home
Page").WebButton("Submit").SetTOProperty("Name", "my button")
```

Submit Method

Submits a form.

Syntax

WebButton(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a button.

```
Browser("Web Testing").Page("Mercury Tours").WebButton("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebButton(*description***).Object.***Method_to_activate***()**
or

myObj=WebButton(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebButton object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/INPUT_button.asp

Example

In the following example, suppose the **Click** method is supported for a button. Use the following statements to activate the **Click** method:

```
Dim MyButton
Set MyButton=Browser("Tibhar-HomePage").Page("Tibhar-
HomePage").Frame("inhalt").WebButton("add to cart").Object
MyButton.Click
```

WebCheckBox Object

A check box with an "ON" and "OFF" state.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- MouseOver Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebCheckBox(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a check box entitled "Roundtrip Ticket".

```
Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").Check Checkpoint("Roundtrip Ticket")
```

Click Method

Clicks an object.

Syntax

WebCheckBox(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "roundtrip" check box.

```
Browser("Mercury Tours").Page("Find Flights").WebCheckBox("roundtrip").Click
```

Exist Method

Checks that an object exists.

Syntax

WebCheckBox(*description***).Exist(**[*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "roundtrip" check box. If the object exists a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").Exist Then
  MsgBox "The check box exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebCheckBox(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the “Event List,” on page 670.

Example

The following example uses the **FireEvent** method to simulate clicking the mouse on a check box.

```
Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").FireEvent "onclick"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebCheckBox(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the check box's type property from the Object Repository.

```
CheckBoxType = Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").GetTOProperty("Type")
' CheckBoxType contains "checkbox"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebCheckBox(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a check box object.

```
Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").MouseOver
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebCheckBox(*description***).GetROProperty(***Property, in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the state of a check box object.

```
CheckState = Browser("Web Testing").Page("Table
Sample").WebCheckBox("SpecialMeal_on").GetROProperty
("Value")
' CheckState contains "OFF"
```

Set Method

Sets an object's value.

Syntax

WebCheckBox(*description***).Set** *value*

Argument	Type	Description
value	String	New value of the object.

Example

In the following example, the **Set** method is used to select the "Roundtrip Tickets" check box.

```
Browser("Mercury Tours").Page("Find Flights").WebCheckBox("roundtrip").Set
"ON"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebCheckBox(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of a check box in run-time.

```
Browser("Mercury Tours").Page("Find
Flights").WebCheckBox("roundtrip").SetTOProperty("Name", "roundtrip")
```

Submit Method

Submits a form.

Syntax

WebCheckBox(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a check box.

```
Browser("Web Testing").Page("Mercury
Tours").WebCheckBox("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebCheckBox(*description***).Object.***Method_to_activate***()**
or

myObj=WebCheckBox(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebCheckBox object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/INPUT_checkbox.asp

Example

In the following example, suppose the **Select** method is supported for a check box. Use the following statements to activate the **Select** method:

```
Dim MyCheckBox
Set MyCheckBox=Browser("Mercury Tours").Page("Method of
Payment").WebCheckBox("saveCC").Object
MyCheckBox.Select
```

WebEdit Object

An edit field, usually contained inside a form.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- GetTOProperty Method
- FireEvent Method
- GetROProperty Method
- MouseOver Method
- Set Method
- SetTOProperty Method
- SetSecure Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebEdit(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following example illustrates the use of the **Check** method on a checkpoint for an WebEdit object entitled "Username".

```
Browser("Mercury Tours").Page("Mercury Tours).WebEdit("Username").Check
Checkpoint("Username")
```

Click Method

Clicks an object.

Syntax

WebEdit(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on a WebEdit object.

```
Browser("Mercury Tours").Page("Mercury Tours).WebEdit("Username").Click
```

Exist Method

Checks that an object exists.

Syntax

WebEdit(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "username" WebEdit object. If the it exists, a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Mercury Tours).WebEdit("Username").Exist
Then
  MsgBox "The edit box exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebEdit(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate moving the mouse over a WebEdit object.

```
Browser("Mercury Tours").Page("Mercury
Tours).WebEdit("Username").FireEvent "onmouseover"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebEdit(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the edit field's name property from the Object Repository.

```
WebEditName = Browser("Mercury Tours").Page("Mercury
Tours).WebEdit("Username").GetTOProperty("Name")
' WebEditName contains "Username"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebEdit(*description***).GetROProperty(***Property***, *in_PropData*)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the type of a WebEdit object.

```
CheckType = Browser("Mercury Tours").Page("Mercury
Tours).WebEdit("Username").GetROProperty("Type")
' CheckType contains "Text"
```

MouseOver Method

Moves the mouse over an object.

Syntax

WebEdit(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a WebEdit object.

```
Browser("Mercury Tours").Page("Mercury
Tours).WebEdit("Username").MouseOver
```

Set Method

Sets an object's value.

Syntax

WebEdit(*description***).Set** *value*

Argument	Type	Description
value	String	New value of the object.

Example

In the following example, the **Set** method is used to insert the desired number of passengers in the "Number of Passengers" edit field.

```
Browser("Mercury Tours").Page("Find Flights").WebEdit("numPassengers").Set
"4"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebEdit(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of a WebEdit object's description in run-time.

```
Browser("Mercury Tours").Page("Mercury
Tours).WebEdit("Username").SetTOProperty("Name", "myusername")
```

SetSecure Method

Sets an object's value in encrypted form. This method is used for edit fields in which a password is entered.

Syntax

WebEdit(*description***).SetSecure** *value*

Argument	Type	Description
value	String	New value of the object in encrypted form.

The following example uses the **SetSecure** method to set an edit field's value in encrypted form.

```
Browser("Mercury Tours").Page("Mercury Tours").WebEdit("username").Set
"mercury"
Browser("Mercury Tours").Page("Mercury
Tours").WebEdit("password").SetSecure "38cf389da9621eb4822cd7"
```

Submit Method

Submits a form.

Syntax

WebEdit(*description***).Submit**

Example

The following example uses the **Submit** method to set a value in an edit field and then submit a form.

```
Browser("Web Testing").Page("Mercury Tours").WebEdit("username").Set
"mercury"
Browser("Web Testing").Page("Mercury Tours").WebEdit("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebEdit(*description***).Object.***Method_to_activate***()**

or

myObj=WebEdit(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebEdit object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/INPUT_text.asp

Example

In the following example, suppose the **focus** method is supported for a edit field. Use the following statements to activate the **focus** method:

```
Dim MyWebEdit
Set MyWebEdit=Browser("Mercury Tours").Page("Mercury
Tours").WebEdit("username").Object
MyWebEdit.focus
```

WebElement Object

A general Web object that can represent any other Web object.

Remarks

This object may be recorded for HTML tags such as DIV and SPAN that are not represented by a specific Web object. This can be useful to check values like font type and color. The WebElement object can also be entered in the test script manually in order to identify an object that QuickTestdoes not recognize automatically.

Note: By default, DIV and SPAN tags are not recorded. You can set QuickTest to record these tags in the Web Event Recording Configuration dialog box. For more information, refer to the *QuickTest User's Guide*.

To add the WebElement object to your script manually, you must provide a unique description for the object using the following general syntax:

WebElement ("*object_identifier1* **:=** *identifier1_value***" , ... , "***object_identifierX* **:=** *identifierX_value***")**

Use as many identifier definitions as necessary to provide a unique description.

Example

QuickTestdoes not record INPUT objects of type: "hidden". The example below uses the WebElement object to find the name of the "hidden" object containing the text: "text of hidden".

```
hidden_name = Browser("mybrowser").Page("mypage").WebElement("html
tag:=INPUT", "outer_html:=text of hidden").GetROProperty("name")
```

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- GetROProperty Method
- MouseOver Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebElement(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a check box entitled "Roundtrip Ticket".

```
Check Checkpoint("Roundtrip Ticket")
```

Click Method

Clicks an object.

Syntax

WebElement(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on a form.

```
Browser("New Page").Page("New Page").WebElement("html tag:=Form").Click
```

Exist Method

Checks that an object exists.

Syntax

WebElement(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the form. If the form exists, a message box appears confirming its appearance.

```
If Browser("New Page").Page("New Page").WebElement("html
tag:=Form").Exist Then
  MsgBox "The form exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebElement(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the “Event List,” on page 670.

Example

The following example uses the **FireEvent** method to simulate a mouse click on a form.

```
Browser("New Page").Page("New Page").WebElement("html
tag:=Form").FireEvent "onclick"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebElement(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the WebElement's htmltag property from the Object Repository.

```
ElementTag = Browser("New Page").Page("New Page").WebElement("just
text").GetTOProperty("htmltag")
'ElementTag contains "SPAN"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebElement(*description***).GetROProperty(***Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the name of a WebElement.

```
ElementName=Browser("New Page").Page("New Page").WebElement("just
text").GetROProperty("Name")
' ElementName contains "just text"
```

MouseOver Method

Moves the mouse over an object.

Syntax

WebElement(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over an element.

```
Browser("New Page").Page("New Page").WebElement("just text").MouseOver
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebElement(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the style of a image's WebElement in run-time.

```
Browser("New Page").Page("New Page").WebElement("just
text").SetTOProperty("Style", "‘color:blue’")
```

Submit Method

Submits a form.

Syntax

WebElement(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a WebElement object.

```
Browser("Web Testing").Page("Mercury
Tours").WebElement("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebElement(*description***).Object.***Method_to_activate***()**
or

myObj=WebElement(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and method of the WebElement object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/html/referen
ce/elements.asp
```

Click on the link that matches your WebElement object

Example

In the following example, suppose the **Refresh** method is supported for a Web element. Use the following statements to activate the **Refresh** method:

```
Dim MyElement
Set MyElement=Browser("New Page").Page("New Page").WebElement("just
text").Object
MyElement.Refresh
```

WebFile Object

An edit field with a "Browse" button attached, used to select a file from the File dialog box.

Associated Methods:

- Click Method
- Exist Method
- FireEvent Method
- GetTOProperty Method
- MouseOver Method
- GetROProperty Method
- Set Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Click Method

Clicks an object.

Syntax

WebFile(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on a WebFile object.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form).WebFile("name-of-files").Click
```

Exist Method

Checks that an object exists.

Syntax

WebFile(*description***).Exist(**[*timeout*]**)**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the WebFile object. If the field exists a message box appears confirming its appearance.

```
If
Browser("Fill-Out-Form").Page("Fill-Out-Form).WebFile("name-of-files").Exist
Then
  MsgBox "the WebFile object exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebFile(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate moving the focus to the WebFile object.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-of-
files").FireEvent "onfocus"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebFile(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the Web file's name property from the Object Repository.

```
WebFileName = Browser("Fill-Out-Form").Page("Fill-Out-
Form").WebFile("name-of-files").GetTOProperty("Name")
' WebFileName contains "name-of-files"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebFile(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a WebFile object.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-of-
files").MouseOver
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebFile(*description***).GetROProperty(***Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the type of a WebFile object.

```
CheckType = Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-
of-files").GetROProperty("type")
' CheckType contains "file"
```

Set Method

Sets an object's value.

Syntax

WebFile(*description***).Set** *value*

Argument	Type	Description
value	String	New value of the object.

Example

In the following example, the **Set** method is used to insert the desired text in the name-of-files WebFile object.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-of-files").Set
"c:\tmp\txt.log"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebFile(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the html id of the WebFile object.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-of-
files").SetTOProperty "html id", 4
```

Submit Method

Submits a form.

Syntax

WebFile(*description***).Submit**

Example

The following example uses the **Submit** method to set a value in a WebFile object and then submit a form.

```
Browser("Fill-Out-Form").Page("Fill-Out-Form").WebFile("name-of-files").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebFile(*description***).Object.***Method_to_activate***()**
or

myObj=WebFile(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebFile object see:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/INPUT_file.asp

Example

In the following example, suppose the **clearAttributes** method is supported for a WebFile object. Use the following statements to activate the **clearAttributes** method:

```
Dim MyWebFile
Set MyWebFile=Browser("Fill-Out Form").Page("Fill-Out
Form").WebFile("name_of_files").Object
MyWebFile.clearAttributes
```

WebList Object

A dropdown box or multiple selection list.

Associated Methods:

- Check Method
- Click Method
- Deselect Method
- Exist Method
- ExtendSelect Method
- FireEvent Method
- GetTOProperty Method
- GetROProperty Method
- MouseOver Method
- Select Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebList(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a checkpoint for a list entitled "depart".

```
Browser("Mercury Tours").Page("Mercury Tours").WebList("depart").Check
Checkpoint("depart")
```

Click Method

Clicks an object.

Syntax

WebList(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "depart" list.

```
Browser("Mercury Tours").Page("Mercury Tours").WebList("depart").Click
```

Deselect Method

Deselects a value from a list.

Syntax

WebList(*description***).Deselect** *value*

Argument	Type	Description
value	String	Value to deselect. The value can either be one of the items in the list or an index. To specify the index of the item to deselect, use "#index".

Example

In the following example, New York is selected from a list of possible departure cities and then deselected from the list using the **Deselect** method.

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select "New
York"
Browser("Mercury Tours").Page("Find Flights").WebList("arrive").Select "Paris"
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Deselect "New
York"
```

Exist Method

Checks that an object exists.

Syntax

WebList(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the arrival list. If the list exists a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Mercury Tours").WebList("arrive").Exist Then
  MsgBox "The arrival list exists."
End If
```

ExtendSelect Method

Adds selected items to a multi-selection list.

Syntax

WebList(*description***).ExtendSelect** *value*

Argument	Type	Description
value	String	Value to add to the selection. The value can either be one of the items in the list or an index. To specify the index of the item to select, use "#index".

Example

The following example uses the **ExtendSelect** method to manually select socks from a list of clothing after jeans has already been selected.

```
Browser("index").Page("Fill-Out Form_2").WebList("what-to-wear").Select
"Jeans"
Browser("index").Page("Fill-Out Form_2").WebList("what-to-
wear").ExtendSelect "Socks"
```

You can also select the item according to its indexed number in the list.

```
Browser("index").Page("Fill-Out Form_2").WebList("what-to-
wear").ExtendSelect "#2"
```

FireEvent Method

Triggers an event.

Syntax

WebList(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate moving the mouse focus from the list.

```
Browser("Mercury Tours").Page("Mercury Tours").WebList("depart").FireEvent
"onblur"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebList(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the list's name property from the Object Repository.

```
ListName = Browser("Mercury Tours").Page("Mercury
Tours").WebList("depart").GetTOProperty("Name")
' ListName contains "depart"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebList(*description***).GetROProperty(***Property***, *in_PropData*)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the value of a list object.

```
ListValue = Browser("Mercury Tours").Page("Find
Flights").WebList("depart").GetROProperty("Value")
' ListValue contains "Value"
```

MouseOver Method

Moves the mouse over an object.

Syntax

WebList(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a list.

```
Browser("Mercury Tours").Page("Mercury Tours").WebList("depart").MouseOver
```

Select Method

Selects a value from the list. Replaces previous selections if they exist.

Syntax

WebList(*description***).Select** *value*

Argument	Type	Description
value	String	Value to select. The value can either be one of the items in the list or an index. To specify the index of the item to select, use "#index". Index values begin with 1.

Example

In the following example, the **Select** method is used to select a city by its name in the list of possible departure cities.

```
Browser("Mercury Tours").Page("Find Flights").WebList("depart").Select
"London"
```

In the following example, the **Select** method is used to select a city by its index number in the list of possible arrival cities.

```
Browser("Mercury Tours").Page("Find Flights_2").WebList("arrive").Select "#4"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebList(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the name of the list's description in run-time.

```
Browser("Mercury Tours").Page("Mercury
Tours").WebList("depart").SetTOProperty("Name", "departure_flights")
```

Submit Method

Submits a form.

Syntax

WebList(*description***).Submit**

Example

The following example uses the **Submit** method to set a value in a dropdown list and then submit a form.

```
Browser("Web Testing").Page("Mercury Tours").WebList("username").Set
"mercury"
Browser("Web Testing").Page("Mercury Tours").WebList("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebList(*description***).Object.***Method_to_activate***()**
or

myObj=WebList(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebList object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/refere
nce/objects/SELECT.asp
```

Example

In the following example, suppose the **Refresh** method is supported for a list. Use the following statements to activate the **Refresh** method:

```
Dim MyList
Set MyList=Browser("Mercury Tours").Page("Mercury
Tours").WebList("depart").Object
MyList.Refresh
```

WebRadioGroup Object

A radio button, usually used to select one of a group of mutually exclusive options.

Associated Methods:

- Check Method
- Click Method
- Exist Method
- FireEvent Method
- GetROProperty Method
- GetTOProperty Method
- MouseOver Method
- Output Method
- Select Method
- SetTOProperty Method
- Submit Method

Associated Properties:

- Object Property

Check Method

Executes a checkpoint.

Syntax

WebRadioGroup(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a radio button for a radio button entitled "seat pref".

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").Check Checkpoint ("seat pref")
```

Click Method

Clicks an object.

Syntax

WebRadioGroup(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the radio button at location 3, 7.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").Click 3, 7
```

Exist Method

Checks that an object exists.

Syntax

WebRadioGroup(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "seat pref" radio button group. If the radio buttons exist, a message box appears confirming their appearance.

```
If Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").Exist Then
  MsgBox "The radio button group exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebRadioGroup(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the “Event List,” on page 670.

Example

The following example uses the **FireEvent** method to simulate clicking the mouse on a radio button.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").FireEvent "onclick"
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebRadioGroup(*description***).GetROProperty(***Property***, *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the GetROProperty method to retrieve the number of items in a WebRadioGroup.

```
NumOfItems = Browser("Mercury Tours").Page("Find
Flights").WebRadioGroup("seat pref").GetROProperty("Items Count")
' NumOfItems contains "3"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebRadioGroup(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the radio button's name property from the Object Repository.

```
RadioName = Browser("Mercury Tours").Page("Find
Flights").WebRadioGroup("seat pref").GetTOProperty("Name")
' RadioName contains "seat pref"
```

Note: GetTOProperty differs from the **GetROProperty** method, which returns the value from the browser in run-time. **GetTOProperty** returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebRadioGroup(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a RadioGroup object.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").MouseOver
```

Output Method

Inserts the value of an object into an output parameter.

Syntax

WebRadioGroup(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output parameter.

Example

The following example uses the **Output** method to place a text item into an output parameter entitled *seat pref*.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").Output Checkpoint("seat pref")
```

Select Method

Selects a radio button from the specified radio group. Replaces previous selections if they exist.

Syntax

WebRadioGroup(*description***).Select** *value*

Argument	Type	Description
value	String	Value to select. The value can either be one of the items in the list or an index. To specify the index of the item to select, use "#index". Index values begin with 1.

Example

In the following example, the **Select** method is used to select the "Window" item in a Web radio group in the seating preference section of the Mercury Tours application.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").Select "Window"
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebRadioGroup(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to select the first radio button in run-time.

```
Browser("Mercury Tours").Page("Find Flights").WebRadioGroup("seat
pref").SetTOProperty("selected item index", "1")
```

Submit Method

Submits a form.

Syntax

WebRadioGroup(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a radio button.

```
Browser("Web Testing").Page("Mercury
Tours").WebRadioGroup("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

WebRadioGroup(*description***).Object.***Method_to_activate***()**
or

myObj=WebRadioGroup(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebRadioGroup object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/refere
nce/objects/INPUT_radio.asp
```

Example

In the following example, suppose the **Click** method is supported for your radio button. Use the following statements to activate the **Click** method:

```
Dim MyRadio
Set MyRadio=Browser("Mercury Tours").Page("Find
Flights").WebRadioGroup("seatPref").Object
MyRadio.Click
```

WebTable Object

A table containing a variable number of rows and columns.

Associated Methods:

- Check Method
- ChildItem Method
- ChildItemCount Method
- Click Method
- ColumnCount Method
- Exist Method
- FireEvent Method
- GetCellData Method
- GetROProperty Method
- GetTOProperty Method
- MouseOver Method
- Output Method
- RowCount Method
- SetTOProperty Method
- Submit Method

Associated Properties:

➤ Object Property

Check Method

Executes a checkpoint.

Syntax

WebTable(*description***).Check checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the checkpoint.

Return Value

Boolean

Example

The following illustrates the use of the **Check** method on a table for a result entitled "Outbound Flight".

```
Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlight").Check Checkpoint("Outbound Flight")
```

ChildItem Method

Returns a test object from the cell by type and index.

Syntax

WebTable(*description***).ChildItem(***row***, *column*, *MicClass*, *index*)**

Argument	Type	Description
row	Number	Row number where the cell is located. The first row in the table is 1.
column	Number	Column number where the cell is located. The first Column in the table is 1.
MicClass	String	Object type.
index	Number	Index of the object of type MicClass in the cell. It is used to specify the desired element when there is more then one object of type MicClass in the cell. The first object has an index of 0.

Return Value

Object

Example

The following example uses the **ChildItem** method to set the second edit box from the sample table to "Example".

```
Set WebEditObj = Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").ChildItem(8, 2, "WebEdit", 0)
WebEditObj.Set "Example"
```

ChildItemCount Method

Returns the number of objects of a specific type in a cell.

Syntax

WebTable(*description***).ChildItemCount(***row***,** *column***,** *MicClass***)**

Argument	Type	Description
row	Number	Row number where the cell is located. The first row in the table is 1.
column	Number	Column number where the cell is located. The first Column in the table is 1.
MicClass	String	Object type.

Return Value

Number

Example

The following example uses the **ChildItemCount** method to count the number of edit fields in a cell in the sample table.

```
Dim NumEdit
Set NumEdit = Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").ChildItemCount(1, 2, "WebEdit")
‘NumEdit contains 1.
```

Click Method

Clicks an object.

Syntax

WebTable(*description***).Click** [*x*, *y*]

Argument	Type	Description
x	Number	**Optional.** The x location of the click.
y	Number	**Optional.** The y location of the click.

Example

In the following example, the **Click** method is executed on the "FirstName" table.

```
Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").Click
```

ColumnCount Method

Return number of columns in a specified row.

Syntax

WebTable(*description***).ColumnCount(***row***)**

Argument	Type	Description
row	Number	Row number. The first row in the table is 1.

Return Value

Number

Example

The following example uses the **ColumnCount** method to determine the number of columns in the first row of the "Sample Table," on page 670.

```
Dim NumColumns
Set NumColumns = Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlights").ColumnCount(1)
  MsgBox "The number of columns in the row is " & numcolumns
```

Exist Method

Checks that an object exists.

Syntax

WebTable(*description***).Exist([***timeout***])**

Argument	Type	Description
timeout	String	**Optional.** Specifies the length of time to search for the object.

Return Value

Boolean

Example

The following example uses the **Exist** method to determine the existence of the "OutBoundFlights" table. If the table exists a message box appears confirming its appearance.

```
If Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlights").Exist Then
  MsgBox "The table exists."
End If
```

FireEvent Method

Triggers an event.

Syntax

WebTable(*description***).FireEvent** *value*

Argument	Type	Description
value	String	The Name of event to trigger. For the list of events, see the "Event List," on page 670.

Example

The following example uses the **FireEvent** method to simulate clicking the mouse on a table.

```
Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").FireEvent "onclick"
```

GetCellData Method

Returns the text for a specified cell.

Syntax

WebTable(*description***).GetCellData(***row***,** *column***)**

Argument	Type	Description
row	Number	Row number where the cell is located. The first row in the table is 1.
column	Number	Column number where the cell is located. The first column in the table is 1.

Return Value

String

Example

In the following example the **GetCellData** method is used to place the contents of the cell located in cell 1, column 1 into a comment.

```
Dim text
Set text = Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlight").GetCellData(1, 1)
  MsgBox "text contains" &text
```

GetROProperty Method

Returns the current value of a property for an item. The value is taken from the run-time object.

Syntax

WebTable(*description***).GetROProperty(***Property***,** *in_PropData***)**

Argument	Type	Description
Property	String	Property to retrieve from the object.
in_PropData	N/A	Not in use at this time.

Return Value

Variant

Example

The following example uses the **GetROProperty** method to retrieve the name of a table.

```
TableName = Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").GetROProperty("name")
' TableName contains "FirstName"
```

GetTOProperty Method

Returns the specified value of a property for an object. The value is taken from the Object Repository.

Syntax

WebTable(*description***).GetTOProperty(***Property***)**

Argument	Type	Description
Property	String	Property to retrieve from the object description.

Return Value

Variant

Example

The following example uses the **GetTOProperty** method to retrieve the value of the WebTable's HtmlTag property from the Object Repository.

```
Dim TableTag
Set TableTag = Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").GetTOProperty("HtmlTag")
' TableTag contains "TABLE"
```

Note: GetTOProperty differs from the GetROProperty method, which returns the value from the browser in run-time. GetTOProperty returns the value from the object's description as set in the Object Repository.

MouseOver Method

Moves the mouse over an object.

Syntax

WebTable(*description***).MouseOver**

Example

The following example uses the **MouseOver** method to move the mouse over a table.

```
Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").MouseOver
```

Output Method

Inserts the value of an object into an output parameter.

Syntax

WebTable(*description***).Output Checkpoint(***name***)**

Argument	Type	Description
name	string	The name of the output parameter.

Example

The following example uses the **Output** method to place a text item into an output parameter entitled *Outbound_Flight*.

```
Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlights").Output Checkpoint("Outbound_Flight")
```

RowCount Method

Return number of rows in a table.

Syntax

WebTable(*description***).RowCount**

Return Value

Number

Example

The following example uses the **RowCount** method to determine the number of rows in the "Sample Table," on page 670.

```
Dim NumRows
Set NumRows=Browser("Mercury Tours").Page("Search
Results").WebTable("OutboundFlights").RowCount
' NumRows contains 5
```

SetTOProperty Method

Sets the specified value of a property for an item.

Note: SetTOProperty changes the properties used to identify an object during run-time. It has no effect on the ActiveScreen or the values saved in the object repository for the object.

Syntax

WebTable(*description***).SetTOProperty** *Property, Value*

Argument	Type	Description
Property	String	Property to add to the object description.
Value	Variant	The value to assign to the listed property.

Example

The following example uses the **SetTOProperty** method to set the number of rows in a table in run-time.

```
Browser("Mercury Tours").Page("Method of
Payment").WebTable("FirstName").SetTOProperty "rows", 5
```

Submit Method

Submits a form.

Syntax

WebTable(*description***).Submit**

Example

The following example uses the **Submit** method to submit a form from a table.

```
Browser("Web Testing").Page("Mercury Tours").WebTable("username").Submit
```

Object Property

Accesses the internal methods and properties of an object. For more information, refer to "The Object Property," on page 671.

Syntax

Browser(*description***).Object.***Method_to_activate***()**
or

myObj=Browser(*description***).Object**
myObj.*Method_to_activate***()**

For a list of properties and methods of the WebTable object see:

```
http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/TABLE.asp
```

Example

In the following example, suppose the **blur** method is supported for a table. Use the following statements to activate the **blur** method:

```
Dim MyTable
Set MyTable=Browser("Mercury Tours").Page("Method of
Payment").WebTable("firstName").Object
MyTable.blur
```

Referenced Information

Sample Table

Search Results
FLIGHTS

Flights Departing from New York to Paris on 06/05/01

Flight	Depart Time	Cost
Blue Sky Air 340	8am	$438
Blue Sky Air 341	1pm	$391
Blue Sky Air 342	5pm	$414
Blue Sky Air 343	11pm	$359

Event List

Any of the following events can be used as the value for the **FireEvent** method.

onchange, onclick, ondblclick, onblur, onfocus, onmousedown, onmouseup, onmouseover, onmouseout, onsubmit, onreset, onpropertychange

Appendix

The Object Property

The Internet Explorer Document Object Model (DOM) is a set of COM objects that correspond to the elements you see on a Web page. Within Internet Explorer, every HTML element (<IMG>, <A>, <TABLE>, and so on) is programmable via an object that is part of the overall object model.

You can modify the appearance and behavior of an HTML element by altering an object's properties and calling its methods. In addition to methods and properties, the objects also fire events to signal user interaction or changes in the corresponding HTML element.

To allow access to these objects, Internet Explorer creates a top-level document object for each HTML document it displays. This document object represents the entire page. From this document object you can access the rest of the object hierarchy by using properties and collections. For example, you will use the *links* property of the document object to retrieve the actual link collection.

When you use the Object property of a Web Element in your script, you actually get a reference to the DOM object. This means that every action you can perform on the DOM object, you can also perform on the Web Element with the Object property.

Example 1

```
document.location.href =www.mercuryinteractive.com
```

will have the same functionality as:

```
Browser(browser_name).page(page_name).Object.location.href =
www.mercuryinteractive.com
```

because Browser(browser_name).page(page_name).Object is the DOM's document.

Example 2

If you have an ActiveX Control embedded in the HTML file, you can access its properties and activate its methods using the Object Property.

```
Browser("index").Page("Untitled").ActiveX("MSCAL.Calendar.7").Object.Day =
20
```

In this example,
Browser("index").Page("Untitled").ActiveX("MSCAL.Calendar.7").Object is a reference to the calendar ActiveX itself, and Day is its property. You can Set this property (like in the example), or Get it.

Example 3

Suppose you have a the following statement in your script:

```
document.MyForm.MyHiddenField.value = My New Text
```

The following example achieves the same thing by using the Object property, where MyDoc is the DOM's document:

```
Dim MyDoc
Set MyDoc = Browser(browser_name).page(page_name).Object
MyDoc.MyForm.MyHiddenField.value = My New Text
```

Example 4

In this example, LinksCollecton is assigned to the link collection of the page through the Object property. Then, a message box pops for each of the links, with its inner text.

```
Dim LinksCollection,  link
Set LinksCollection = Browser(browser_name).Page(page_name).Object.links
For Each link in LinksCollection
    MsgBox link.innerHTML
Next
```

Additional Information

For more information about the DHTML Document Object Model, refer to:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/om/doc_object.asp

Index

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W

MERCURY INTERACTIVE

Mercury Interactive Corporation
1325 Borregas Avenue
Sunnyvale, CA 94089 USA

Main Telephone: (408) 822-5200
Sales & Information: (800) TEST-911
Customer Support: (877) TEST-HLP
Fax: (408) 822-5300

Home Page: www.mercuryinteractive.com
Customer Support: support.mercuryinteractive.com