CodePanes Collection

Contains the active <u>code panes</u> in the **VBE** object.

Remarks

Use the **CodePanes** collection to access the open code panes in a <u>project</u>.

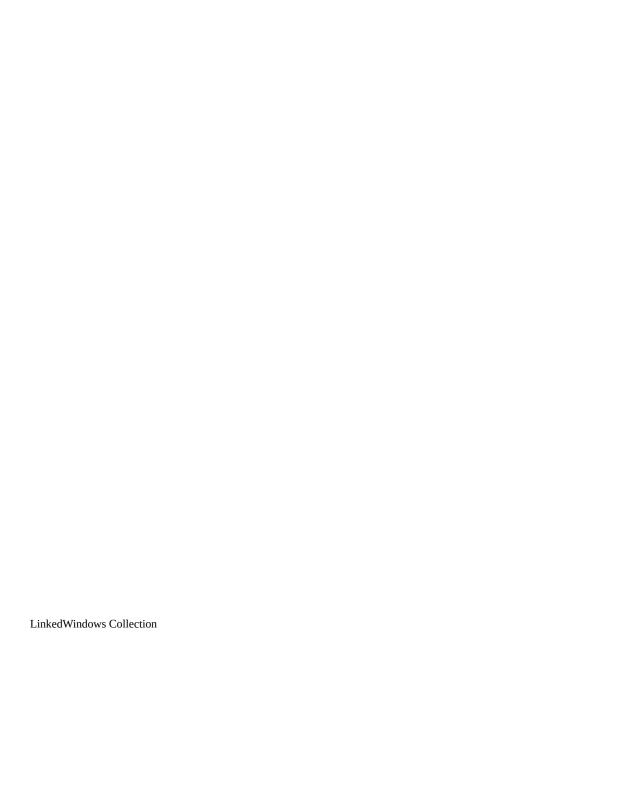
You can use the **Count** property to return the number of active code panes in a <u>collection</u>.

CommandBars Collection

Contains all of the command bars in a <u>project</u>, including command bars that support shortcut menus.

Remarks

Use the **CommandBars** collection to enable <u>add-ins</u> to add command bars and <u>controls</u>, or to add controls to existing, built-in, command bars.



LinkedWindows Collection Window Object

Contains all linked windows in a linked window frame.

Remarks

Use the **LinkedWindows** collection to modify the <u>docked</u> and <u>linked</u> state of windows in the <u>development environment</u>.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

The **LinkedWindowFrame** property of the **Window** object returns a **Window** object that has a valid **LinkedWindows** collection.

Linked window frames contain all windows that can be linked or docked. This includes all windows except code windows, <u>designers</u>, the <u>Object Browser</u> window, and the **Search and Replace** window.

If all the panes from one linked window frame are moved to another window, the linked window frame with no panes is destroyed. However, if all the panes are removed from the main window, it isn't destroyed.

Use the **Visible** property to check or set the visibility of a window.

You can use the **Add** method to add a window to the <u>collection</u> of currently linked windows. A window that is a pane in one linked window frame can be added to another linked window frame. Use the **Remove** method to remove a window from the collection of currently linked windows; this results in the window being unlinked or undocked.

The **LinkedWindows** collection is used to dock and undock windows from the main window frame.

Properties Collection (VBA Add-In Object Model)

Properties Collection Property Object

Represents the properties of an object.

Remarks

Use the **Properties** collection to access the properties displayed in the <u>Properties</u> window. For every property listed in the **Properties** window, there is an object in the **Properties** collection.

References Collection

References Collection Reference Object

Represents the set of references in the <u>project</u>.

Remarks

Use the **References** collection to add or remove references. The **References** collection is the same as the set of references selected in the **References** dialog box.



VBComponents Collection VBComponent Object

Represents the components contained in a project.

Remarks

Use the **VBComponents** collection to access, add, or remove components in a project. A component can be a <u>form</u>, <u>module</u>, or <u>class</u>. The **VBComponents** collection is a standard <u>collection</u> that can be used in a **For Each** block.

You can use the **Parent** property to return the project the **VBComponents** collection is in.

In Visual Basic for Applications, you can use **Import** method to add a component to a project from a file.

VBProjects Collection



Represents all the <u>projects</u> that are open in the <u>development environment</u>.

Remarks

Use the **VBProjects** collection to access specific projects in an instance of the development environment. **VBProjects** is a standard collection that can be used in a **For Each** block.





Contains all open or permanent windows.

Remarks

Use the **Windows** collection to access **Window** objects.

The **Windows** collection has a fixed set of windows that are always available in the <u>collection</u>, such as the <u>Project window</u>, the <u>Properties window</u>, and a set of windows that represent all open code windows and <u>designer</u> windows. Opening a code or designer window adds a new member to the **Windows** collection. Closing a code or designer window removes a member from the **Windows** collection. Closing a permanent <u>development environment</u> window doesn't remove the corresponding object from this collection, but results in the window not being visible.



Occurs when the **OnAction** <u>property</u> of a corresponding command bar control is set.

Syntax

Sub object_Click (ByVal ctrl As Object, ByRef handled As Boolean, ByRef canceldefault As Boolean)

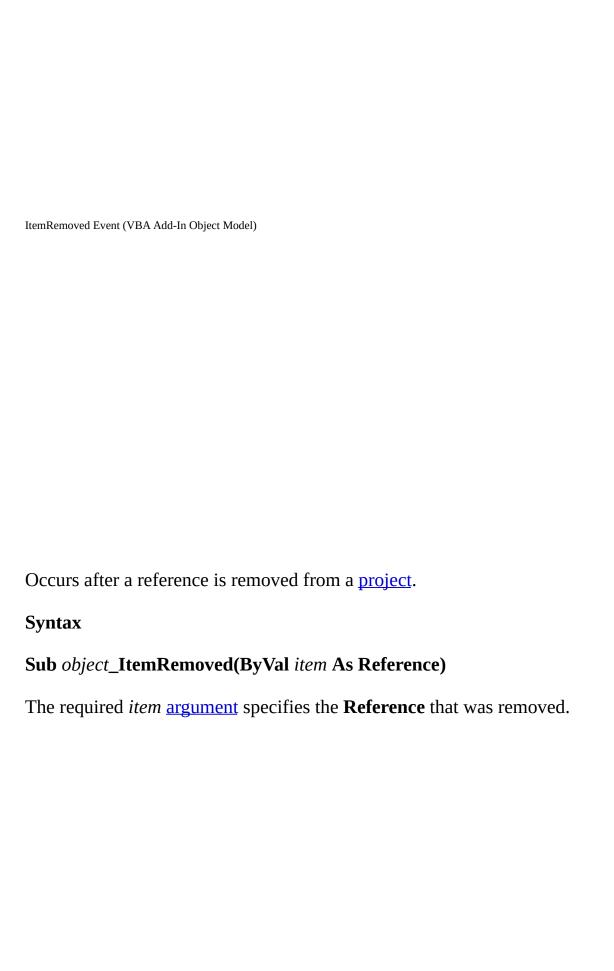
The Click event syntax has these <u>named arguments</u>:

Part	Description
ctrl	Required; Object. Specifies the object that is the source of the Click event.
handled	Required; <u>Boolean</u> . If True , other <u>add-ins</u> should handle the event. If False , the action of the command bar item has not been handled.
canceldefault	Required; Boolean . If True , default behavior is performed unless canceled by a downstream add-in. If False , default behavior is not performed unless restored by a downstream add-in.

Remarks

The Click event is specific to the **CommandBarEvents** object. Use a <u>variable</u> declared using the **WithEvents** keyword to receive the Click event for a **CommandBar** control. This variable should be set to the return value of the **CommandBarEvents** property of the **Events** object. The **CommandBarEvents** property takes the **CommandBar** control as an <u>argument</u>. When the **CommandBar** control is clicked (for the variable you declared using the **WithEvents** keyword), the code is executed.

ItemAdded Event (VBA Add-In Object Model)
Occurs after a reference is added.
Syntax
Sub object_ItemAdded(ByVal item As Reference)
The required <i>item</i> argument specifies the item that was added.
Remarks
The ItemAdded event occurs when a Reference is added to the References collection.





Adds an object to a collection.

Syntax

object.Add(component)

The **Add** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
component	Required. For the LinkedWindows collection, an object. For the VBComponents collection, an enumerated <u>constant</u> representing a <u>class module</u> , a form, or a <u>standard module</u> .
	For the VBProjects collection, an enumerated constant representing a project type.

You can use one of the following constants for the *component* argument:

Constant	Description
<pre>vbext_ct_ClassModule Adds a class module to the collection.</pre>	
vbext_ct_MSForm	Adds a form to the collection.
vbext_ct_StdModule	Adds a standard module to the collection.
vbext_pt_StandAlone	Adds a standalone project to the collection.

Remarks

For the **LinkedWindows** collection, the **Add** method adds a window to the

collection of currently linked windows.

Note You can add a window that is a pane in one <u>linked window frame</u> to another linked window frame; the window is simply moved from one pane to the other. If the linked window frame that the window was moved from no longer contains any panes, it's destroyed.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements generate run-time errors when run on the Macintosh.

For the **VBComponents** collection, the **Add** method creates a new standard component and adds it to the <u>project</u>.

For the **VBComponents** collection, the **Add** method returns a **VBComponent** object. For the **LinkedWindows** collection, the **Add** method returns **Nothing**.

For the **VBProjects** collection, the **Add** method returns a **VBProject** object and adds a project to the **VBProjects** collection.



Syntax

object.AddFromFile(filename)

The **AddFromFile** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
filename	Required. A <u>string expression</u> specifying the name of the file you want to add to the project or module. If the file name isn't found and a path name isn't specified, the directories searched by the Windows OpenFile function are searched.

Remarks

For the **CodeModule** object, the **AddFromFile** method inserts the contents of the file starting on the line preceding the first <u>procedure</u> in the <u>code module</u>. If the module doesn't contain procedures, **AddFromFile** places the contents of the file at the end of the module.



Syntax

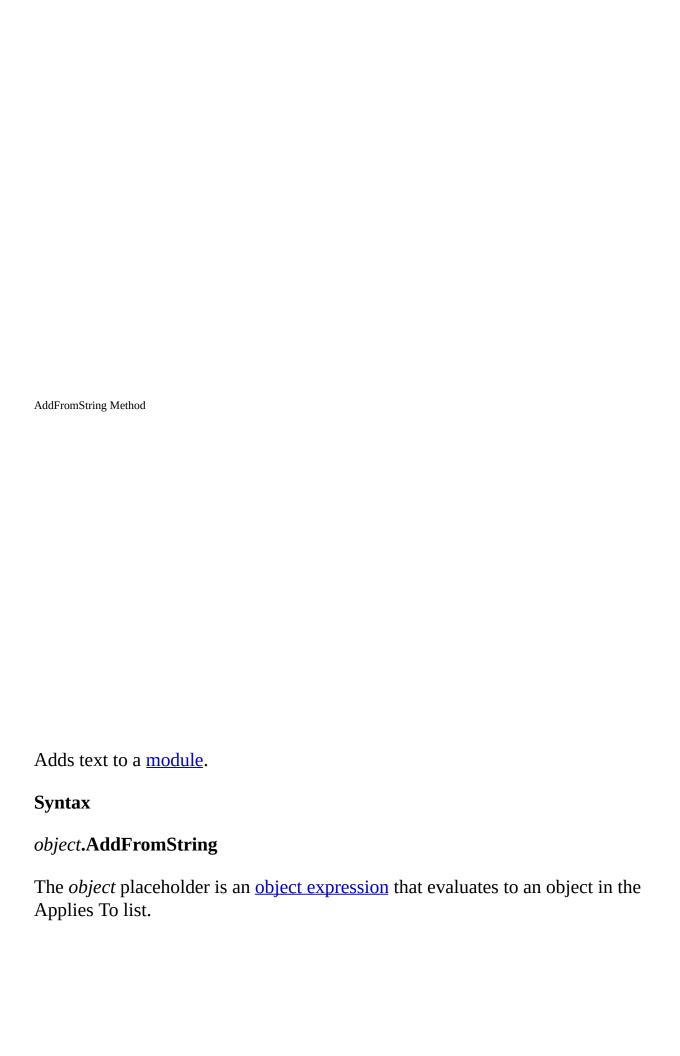
object.AddFromGuid(guid, major, minor) As Reference

The **AddFromGuid** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
guid	Required. A <u>string expression</u> representing the GUID of the reference.
major	Required. A <u>Long</u> specifying the major version number of the reference.
minor	Required. A Long specifying the minor version number of the reference.

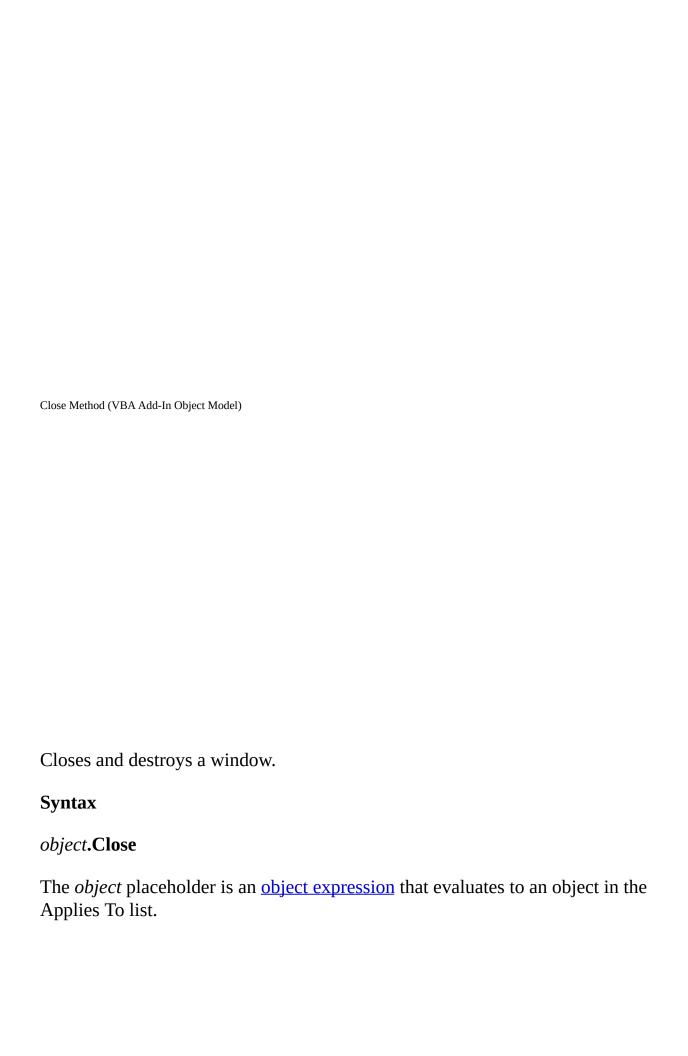
Remarks

The **AddFromGuid** method searches the <u>registry</u> to find the reference you want to add. The GUID can be a <u>type library</u>, <u>control</u>, class identifier, and so on.



Remarks

The **AddFromString** method inserts the text starting on the line preceding the first <u>procedure</u> in the module. If the module doesn't contain procedures, **AddFromString** places the text at the end of the module.



Remarks

The following types of windows respond to the **Close** method in different ways:

For a window that is a <u>code pane</u>, **Close** destroys the code pane.

For a window that is a <u>designer</u>, **Close** destroys the contained designer.

For windows that are always available on the **View** menu, **Close** hides the window.

CreateEventProc Method Creates an event <u>procedure</u>.

Syntax

object.CreateEventProc(eventname, objectname) As Long

The **CreateEventProc** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
eventname	Required. A <u>string expression</u> specifying the name of the event you want to add to the <u>module</u> .
objectname	Required. A string expression specifying the name of the object that is the source of the event.

Remarks

Use the **CreateEventProc** method to create an event procedure. For example, to create an event procedure for the Click event of a **Command Button** control named Command1 you would use the following code, where CM represents a object of type **CodeModule**:

The **CreateEventProc** method returns the line at which the body of the event procedure starts. **CreateEventProc** fails if the <u>arguments</u> refer to a nonexistent event.

DeleteLines Method

Deletes a single line or a specified range of lines.

Syntax

object.DeleteLines (startline [, count])

The **DeleteLines** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
startline	Required. A Long specifying the first line you want to delete.
count	Optional. A Long specifying the number of lines you want to delete.

Remarks

If you don't specify how many lines you want to delete, **DeleteLines** deletes one line.

Export Method (VBA Add-In Object Model)

Saves a component as a separate file or files.

Syntax

object.Export(filename)

The **Export** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
filename	Required. A <u>String</u> specifying the name of the file that you want to export the component to.

Remarks

When you use the **Export** method to save a component as a separate file or files,

use a file name that doesn't already exist; otherwise, an error occurs.



Syntax

object.**Find(**target, startline, startcol, endline, endcol [, wholeword] [, matchcase] [, patternsearch]) **As Boolean**

The **Find** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
target	Required. A <u>String</u> containing the text or pattern you want to find.
startline	Required. A Long specifying the line at which you want to start the search; will be set to the line of the match if one is found. The first line is number 1.
startcol	Required. A Long specifying the column at which you want to start the search; will be set to the column containing the match if one is found. The first column is 1.
endline	Required. A Long specifying the last line of the match if one is found. The last line may be specified as -1 .
endcol	Required. A Long specifying the last line of the match if one is found. The last column may be designated as -1 .
wholeword	Optional. A <u>Boolean</u> value specifying whether to only match whole words. If True , only matches whole words. False is the default.
matchcase	Optional. A Boolean value specifying whether to match case. If True , the search is case sensitive. False is the default.
patternsearch	Optional. A Boolean value specifying whether or not the target string is a regular expression pattern. If True , the target string is a regular expression pattern. False is the default.

Remarks

Find returns **True** if a match is found and **False** if a match isn't found.

The *matchcase* and *patternsearch* <u>arguments</u> are mutually exclusive; if both

arguments are passed as True, an error occurs.

The content of the **Find** dialog box isn't affected by the **Find** method.

The specified range of lines and columns is inclusive, so a search can find the pattern on the specified last line if *endcol* is supplied as either –1 or the length of the line.

GetSelection Method Returns the selection in a <u>code pane</u>.

Syntax

object.GetSelection(startline, startcol, endline, endcol)

The **GetSelection** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
startline	Required. A <u>Long</u> that returns a value specifying the first line of the selection in the code pane.
startcol	Required. A Long that returns a value specifying the first column of the selection in the code pane.
endline	Required. A Long that returns a value specifying the last line of the selection in the code pane.
endcol	Required. A Long that returns a value specifying the last column of the selection in the code pane.

Remarks

When you use the **GetSelection** method, information is returned in output <u>arguments</u>. As a result, you must pass in <u>variables</u> because the variables will be modified to contain the information when returned.



Syntax

object.Import(filename) As VBComponent

The **Import** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
filename	Required. A <u>String</u> specifying path and file name of the component that you want to import the component from.

Remarks

You can use the **Import** method to add a component, <u>form</u>, <u>module</u>, <u>class</u>, and so on, to your project.

InsertLines Method	
Inserts a line or lines of code at a specified location in a block of code.	
Syntax	
object.InsertLines(line, code)	
The InsertLines syntax has these parts:	
Part Description	

object Required. An object expression that evaluates to an object in the

Applies To list.

line Required. A Long specifying the location at which you want to

insert the code.

code Required. A <u>String</u> containing the code you want to insert.

Remarks

If the text you insert using the **InsertLines** method is carriage return—linefeed delimited, it will be inserted as consecutive lines.



object.Item(index)

The **Item** method syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
index	Required. An expression that specifies the position of a member of the collection. If a <u>numeric expression</u> , <i>index</i> must be a number from 1 to the value of the collection's Count property. If a <u>string expression</u> , <i>index</i> must correspond to the <i>key</i> <u>argument</u> specified when the member was added to the collection.

The following table lists the collections and their corresponding *key* arguments for use with the **Item** method. The string you pass to the **Item** method must match the collection's *key* argument.

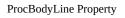
Collection	Key argument
Windows	Caption property setting
LinkedWindows	Caption property setting
CodePanes	No unique string is associated with this collection.
VBProjects	Name property setting
VBComponents	Name property setting
References	Name property setting
Properties	Name property setting

Remarks

The *index* argument can be a numeric value or a string containing the title of the object.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for

compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.



Returns the first line of a <u>procedure</u>.

Syntax

object.ProcBodyLine(procname, prockind) As Long

The **ProcBodyLine** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
procname	Required. A String containing the name of the procedure.
prockind	Required. Specifies the kind of procedure to locate. Because property procedures can have multiple representations in the module, you must specify the kind of procedure you want to locate. All procedures other than property procedures (that is, Sub and Function procedures) use vbext_pk_Proc .

You can use one of the following constants for the *prockind* argument:

Constant	Description
vbext_pk_Get	Specifies a procedure that returns the value of a property.
vbext_pk_Let	Specifies a procedure that assigns a value to a property.
vbext_pk_Set	Specifies a procedure that sets a reference to an object.
vbext_pk_Proc	Specifies all procedures other than property procedures.

Remarks

The first line of a procedure is the line on which the **Sub**, **Function**, or **Property** statement appears.



Returns the number of lines in the specified <u>procedure</u>.

Syntax

object.ProcCountLines(procname, prockind) As Long

The **ProcCountLines** syntax has these parts:

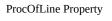
Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
procname	Required. A String containing the name of the procedure.
prockind	Required. Specifies the kind of procedure to locate. Because property procedures can have multiple representations in the module, you must specify the kind of procedure you want to locate. All procedures other than property procedures (that is,
	Sub and Function procedures) use vbext_pk_Proc .

You can use one of the following constants for the *prockind* argument:

Constant	Description
vbext_pk_Get	Specifies a procedure that returns the value of a property.
vbext_pk_Let	Specifies a procedure that assigns a value to a property.
vbext_pk_Set	Specifies a procedure that sets a reference to an object.
vbext_pk_Proc	Specifies all procedures other than property procedures.

Remarks

The **ProcCountLines** property returns the count of all blank or comment lines preceding the procedure declaration and, if the procedure is the last procedure in a <u>code module</u>, any blank lines following the procedure.



Returns the name of the <u>procedure</u> that the specified line is in.

Syntax

object.ProcOfLine(line, prockind) As String

The **ProcOfLine** syntax has these parts:

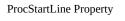
Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
line	Required. A Long specifying the line to check.
prockind	Required. Specifies the kind of procedure to locate. Because property procedures can have multiple representations in the module, you must specify the kind of procedure you want to locate. All procedures other than property procedures (that is, Sub and Function procedures) use vbext_pk_Proc .

You can use one of the following constants for the *prockind* argument:

Constant	Description
vbext_pk_Get	Specifies a procedure that returns the value of a property.
vbext_pk_Let	Specifies a procedure that assigns a value to a property.
vbext_pk_Set	Specifies a procedure that sets a reference to an object.
vbext_pk_Proc	Specifies all procedures other than property procedures.

Remarks

A line is within a procedure if it's a blank line or comment line preceding the procedure declaration and, if the procedure is the last procedure in a <u>code</u> <u>module</u>, a blank line or lines following the procedure.



Returns the line at which the specified <u>procedure</u> begins.

Syntax

object.ProcStartLine(procname, prockind) As Long

The **ProcStartLine** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
procname	Required. A String containing the name of the procedure.
prockind	Required. Specifies the kind of procedure to locate. Because property procedures can have multiple representations in the module, you must specify the kind of procedure you want to locate. All procedures other than property procedures (that is, Sub and Function procedures) use vbext_pk_Proc .

You can use one of the following constants for the *prockind* argument:

Constant	Description
vbext_pk_Get	Specifies a <u>procedure</u> that returns the value of a
	property.
vbext_pk_Let	Specifies a procedure that assigns a value to a property.
vbext_pk_Set	Specifies a procedure that sets a reference to an object.
vbext_pk_Proc	Specifies all procedures other than property procedures.

Remarks

A procedure starts at the first line below the **End Sub** statement of the preceding procedure. If the procedure is the first procedure, it starts at the end of the general Declarations section.



Removes an item from a collection.

Syntax

object.Remove(component)

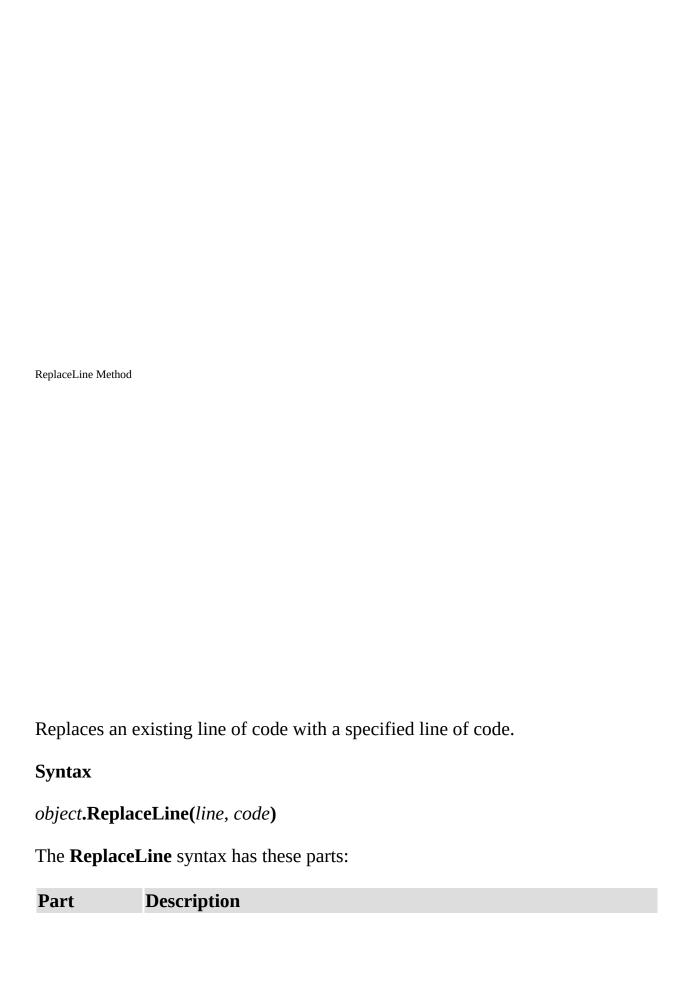
The **Remove** syntax has these parts:

Part	Description
object	Required. An <u>object expression</u> that evaluates to an object in the Applies To list.
component	Required. For the LinkedWindows collection, an object. For the References collection, a reference to a <u>type library</u> or a
	<pre>project. For the VBComponents collection, an enumerated</pre>
	constant representing a class module, a form, or a standard
	module. For the VBProjects collection, a standalone project.

Remarks

When used on the **LinkedWindows** collection, the **Remove** method removes a window from the collection of currently <u>linked windows</u>. The removed window becomes a floating window that has its own <u>linked window frame</u>. The **Remove** method can only be used on a standalone project. It generates a run-time error if you try to use it on a host project.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements generate run-time errors when run on the Macintosh.



object Required. An object expression that evaluates to an object in the

Applies To list.

line Required. A Long specifying the location of the line you want to

replace.

code Required. A <u>String</u> containing the code you want to insert.

SetFocus Method
Managadha Canagadha an ai Cada in dan
Moves the <u>focus</u> to the specified window.
Syntax
object.SetFocus
The <i>object</i> placeholder is an <u>object expression</u> that evaluates to an object in the Applies To list.
Remarks
Use the SetFocus method on windows that are already visible.

SetSelection Method Sets the selection in the <u>code pane</u>. **Syntax** object.SetSelection(startline, startcol, endline, endcol) The **SetSelection** syntax has these parts: Description Part

object	Required. An object expression that evaluates to an object in
	the Applies To list.
startline	Required. A Long specifying the first line of the selection.
startcol	Required. A Long specifying the first column of the selection.
endline	Required. A Long specifying the last line of the selection.
endcol	Required. A Long specifying the last column of the selection.



CodeModule Object

Represents the code behind a component, such as a <u>form</u>, <u>class</u>, or <u>document</u>.

Remarks

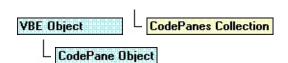
You use the **CodeModule** object to modify (add, delete, or edit) the code associated with a component.

Each component is associated with one **CodeModule** object. However, a **CodeModule** object can be associated with multiple <u>code panes</u>.

The methods associated with the **CodeModule** object enable you to manipulate and return information about the code text on a line-by-line basis. For example, you can use the **AddFromString** method to add text to the <u>module</u>. **AddFromString** places the text just above the first <u>procedure</u> in the module or places the text at the end of the module if there are no procedures.

Use the **Parent** property to return the **VBComponent** object associated with a code module.

CodePane Object



Represents a code pane.

Remarks

Use the **CodePane** object to manipulate the position of visible text or the text selection displayed in the code pane.

You can use the **Show** method to make the code pane you specify visible. Use the **SetSelection** method to set the selection in a code pane and the **GetSelection** method to return the location of the selection in a code pane.

CommandBarEvents Object



Returned by the **CommandBarEvents** property. The **CommandBarEvents** object triggers an event when a <u>control</u> on the command bar is clicked.

Remarks

The **CommandBarEvents** object is returned by the **CommandBarEvents**

property of the **Events** object. The object that is returned has one event in its interface, the Click event. You can handle this event using the **WithEvents** object declaration.



Remarks

The **Events** object provides properties that return <u>event source objects</u>. Use the properties to return event source objects that notify you of changes in the Visual Basic for Applications environment.

The properties of the **Events** object return objects of the same type as the property name. For example, the **CommandBarEvents** property returns the **CommandBarEvents** object.

Property Object

Properties Collection

Property Object

Represents the <u>properties</u> of an object that are visible in the <u>Properties window</u> for any given component.

Remarks

Use **Value** property of the **Property** object to return or set the value of a property of a component.

At a minimum, all components have a **Name** property. Use the **Value** property of the **Property** object to return or set the value of a property. The **Value** property returns a <u>Variant</u> of the appropriate type. If the value returned is an object, the **Value** property returns the **Properties** collection that contains **Property** objects representing the individual properties of the object. You can access each of the **Property** objects by using the **Item** method on the returned **Properties** collection.

If the value returned by the **Property** object is an object, you can use the **Object** property to set the **Property** object to a new object.

Reference Object

References Collection Reference Object

Represents a reference to a type library or a project.

Remarks

Use the **Reference** object to verify whether a reference is still valid.

The **IsBroken** property returns **True** if the reference no longer points to a valid

reference. The **BuiltIn** property returns **True** if the reference is a default reference that can't be moved or removed. Use the **Name** property to determine if the reference you want to add or remove is the correct one.

ReferencesEvents Object
Returned by the ReferencesEvents property. Remarks
The ReferencesEvents object is the source of events that occur when a reference is added to or removed from a <u>project</u> . The ItemAdded event is triggered after a reference is added to a project. The ItemRemoved event is triggered after a reference is removed from a project.



VBComponents Collection VBProject Object

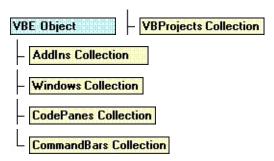
Represents a component, such as a <u>class module</u> or <u>standard module</u>, contained in a <u>project</u>.

Remarks

Use the **VBComponent** object to access the <u>code module</u> associated with a component or to change a component's property settings.

You can use the **Type** property to find out what type of component the **VBComponent** object refers to. Use the **Collection** property to find out what <u>collection</u> the component is in.





The root object that contains all other <u>objects</u> and <u>collections</u> represented in Visual Basic for Applications.

Remarks

You can use the following collections to access the objects contained in the **VBE** object:

Use the **VBProjects** collection to access the collection of <u>projects</u>.

Use the **AddIns** collection to access the collection of add-ins.

Use the **Windows** collection to access the collection of windows.

Use the **CodePanes** collection to access the collection of <u>code panes</u>.

Use the **CommandBars** collection to access the collection of command bars.

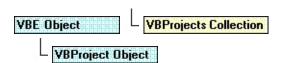
Use the **Events** object to access properties that enable <u>add-ins</u> to connect to all events in Visual Basic for Applications. The properties of the **Events** object

return objects of the same type as the property name. For example, the **CommandBarEvents** property returns the **CommandBarEvents** object.

You can use the **SelectedVBComponent** property to return the active component. The active component is the component that is being tracked in the <u>Project window</u>. If the selected item in the **Project** window isn't a component, **SelectedVBComponent** returns **Nothing**.

Note All objects in this object model have a **VBE** property that points to the **VBE** object.

VBProject Object



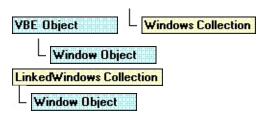
Represents a project.

Remarks

Use the **VBProject** object to set <u>properties</u> for the project, to access the

VBComponents collection, and to access the **References** collection.





Represents a window in the <u>development environment</u>.

Remarks

Use the **Window** object to show, hide, or position windows.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

You can use the **Close** method to close a window in the **Windows** collection. The **Close** method affects different types of windows as follows:

Window	Result of using Close method
Code window	Removes the window from the Windows collection.
<u>Designer</u>	Removes the window from the Windows collection.
Window objects of type <u>linked</u> window frame	Windows become unlinked separate windows.

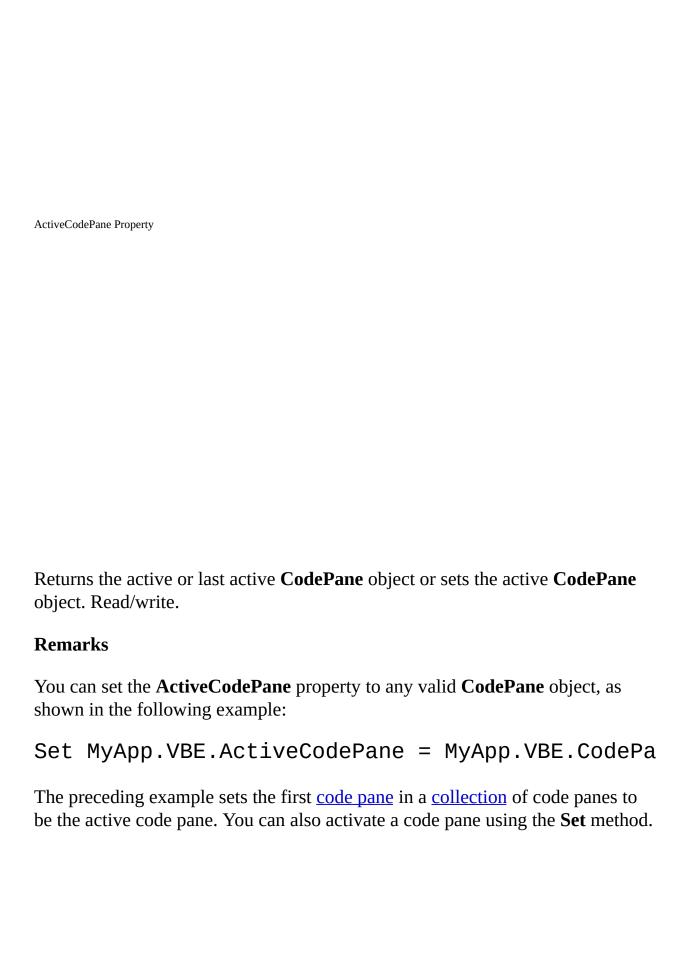
Note Using the **Close** method with code windows and designers actually closes

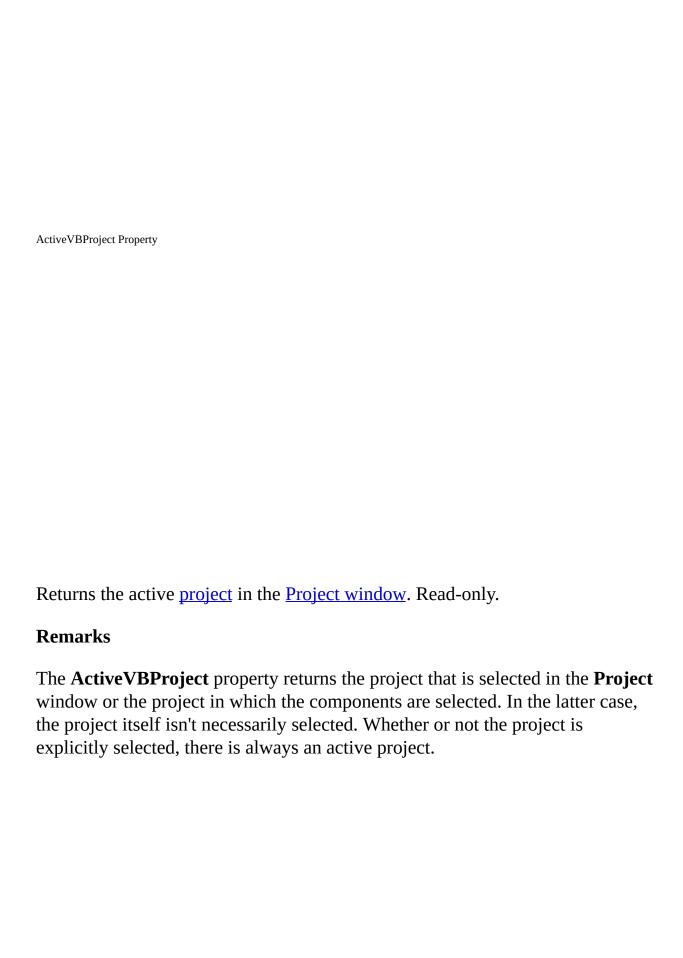
the window. Setting the **Visible** property to **False** hides the window but doesn't close the window. Using the **Close** method with development environment windows, such as the <u>Project window</u> or <u>Properties window</u>, is the same as setting the **Visible** property to **False**.

You can use the **SetFocus** method to move the <u>focus</u> to a window.

You can use the **Visible** property to return or set the visibility of a window.

To find out what type of window you are working with, you can use the **Type** property. If you have more than one window of a type, for example, multiple designers, you can use the **Caption** property to determine the window you're working with. You can also find the window you want to work with using the **DesignerWindow** property of the **VBComponent** object or the **Window** property of the **CodePane** object.





ActiveWindow Property
Returns the active window in the <u>development environment</u> . Read-only.
Remarks
When more than one window is open in the development environment, the ActiveWindow property setting is the window with the <u>focus</u> . If the main window has the focus, ActiveWindow returns Nothing .

BuiltIn Property

Returns a <u>Boolean</u> value indicating whether or not the reference is a default reference that can't be removed. Read-only.

Return Values

The **BuiltIn** property returns these values:

Value	Description
True	The reference is a default reference that can't be
	removed.
False	The reference isn't a default reference; it can be removed.

Caption Property
Returns a <u>String</u> containing the title of the active window. Read-only.
Remarks
The title of the active window is the text displayed in the window's title bar.

CodeModule Property
Returns an object representing the code behind the component. Read-only.
Remarks
The CodeModule property returns Nothing if the component doesn't have a code module associated with it.
represents the code within a component. A component can only have one
component can have several CodePane objects. The CodeModule object represents the code within a component. A component can only have one CodeModule object.

Code	Pane Property
Ret	turns a CodePane object. Read-only.
Re	marks
con	code pane exists, it becomes the active code pane, and the window that tains it becomes the active window. If a code pane doesn't exist for the dule, the CodePane property creates one.



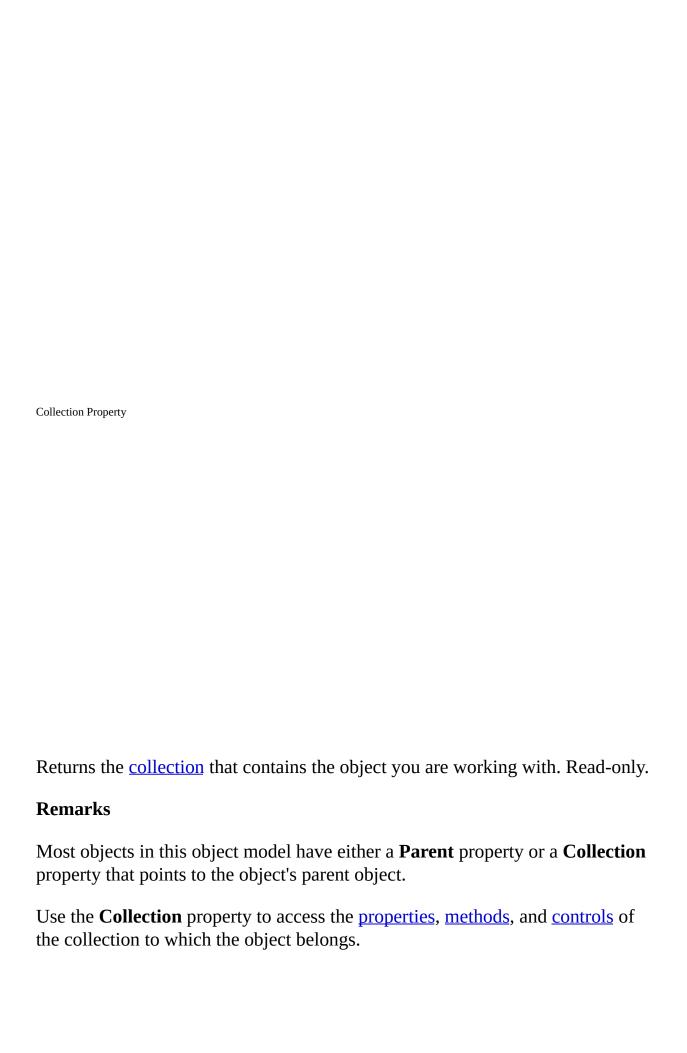
CodePaneView Property

Returns a value indicating whether the <u>code pane</u> is in Procedure view or Full Module view. Read-only.

Return Values

The **CodePaneView** property return values are:

Constant	Description
vbext_cv_ProcedureView	The specified code pane is in Procedure view.
Vbext_cv_FullModuleView	The specified <u>project</u> is in Full Module view.





The cetting for the argument you pass to the CommandRarFvents property is:

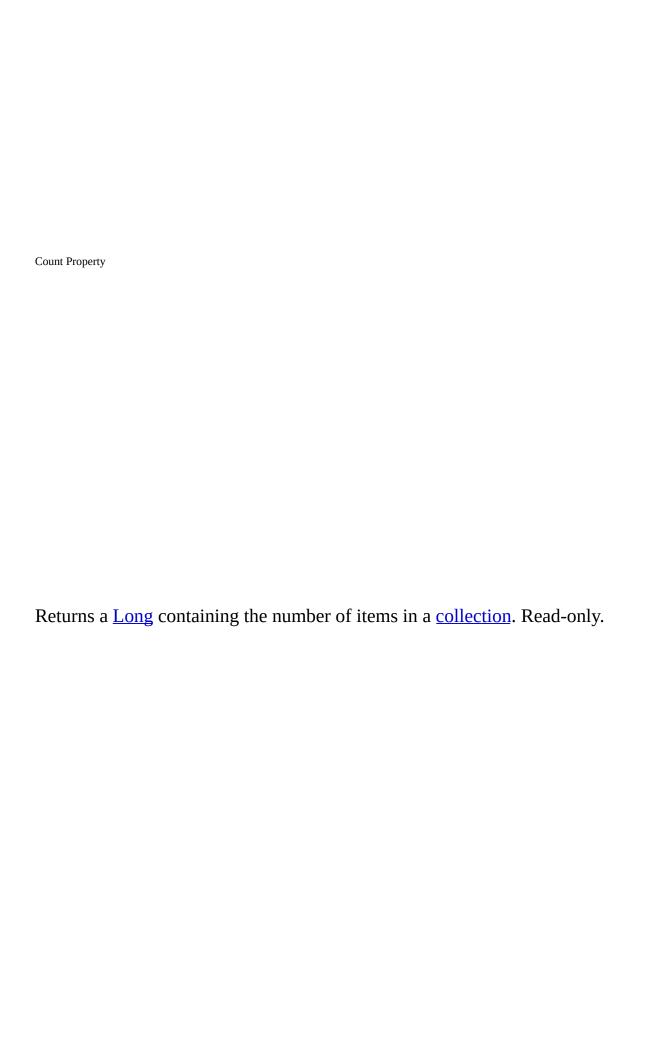
The setting to	or the <u>argume</u>	<u>nt</u> you pass	to the Com	ımanaBarEvents	property is:

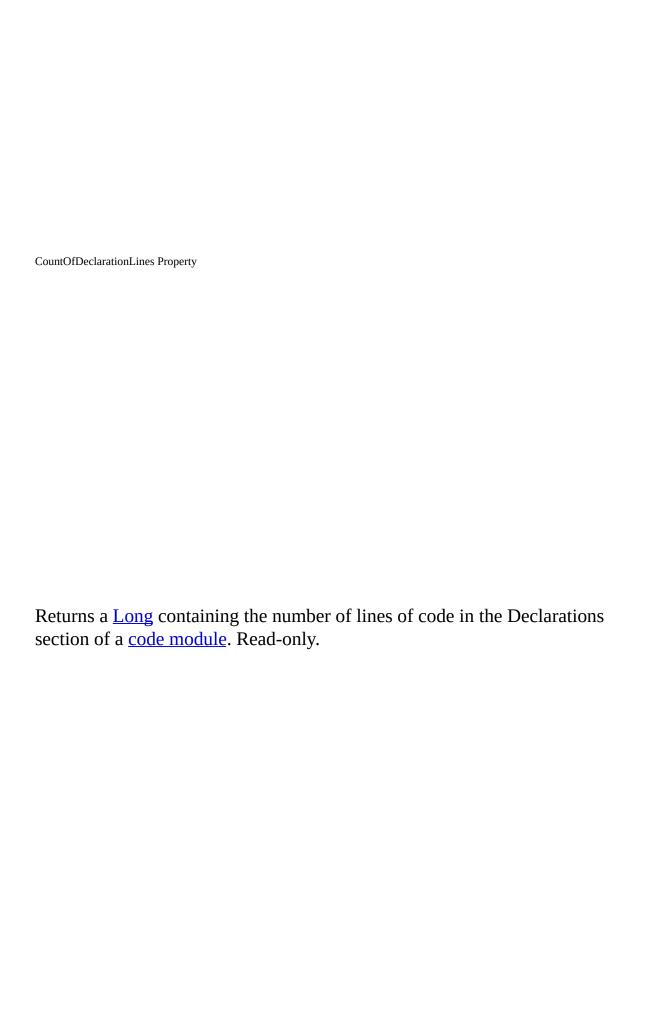
Argument	Description
vbcontrol	Must be an object of type
	CommandBarControl.

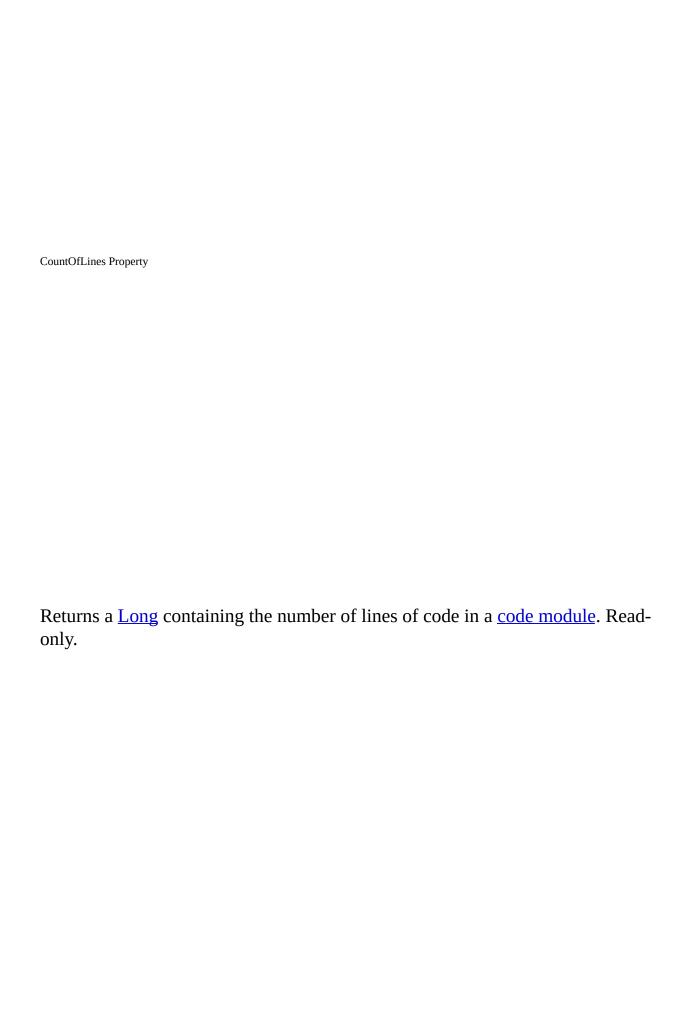
Remarks

Settings

Use the **CommandBarEvents** property to return an <u>event source object</u> that triggers an event when a command bar button is clicked. The argument passed to the **CommandBarEvents** property is the command bar control for which the Click event will be triggered.









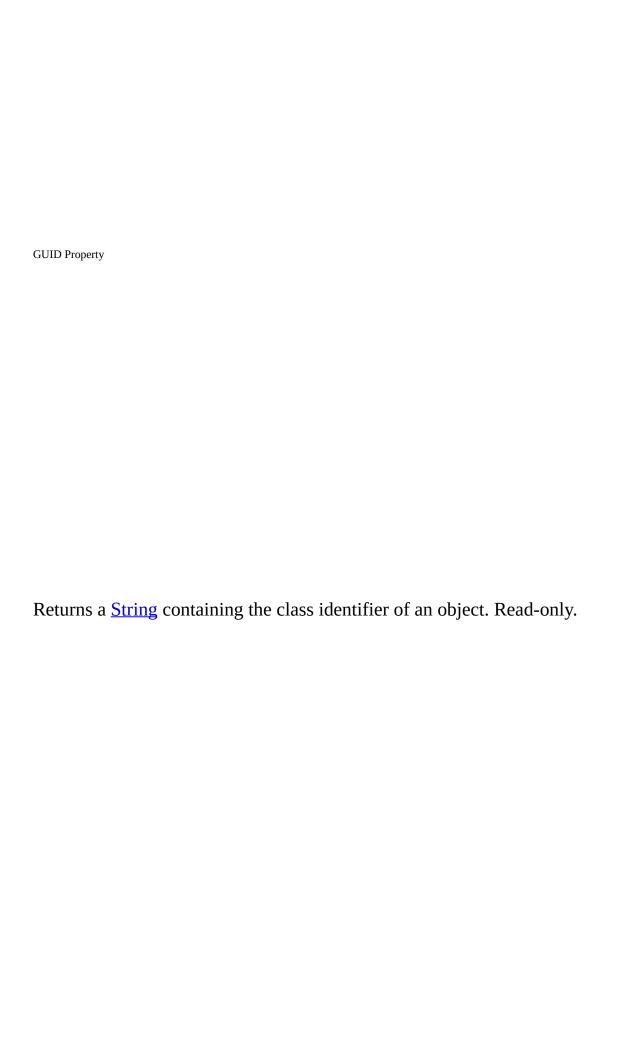
Description Property
Returns or sets a <u>string expression</u> containing a descriptive string associated with an object. For the VBProject object, read/write; for the Reference object, read-
only. Remarks
For the VBProject object, the Description property returns or sets a descriptive string associated with the active <u>project</u> .
For the Reference object, the Description property returns the descriptive name of the reference.

Designer Property
Returns the object that enables you to access the design characteristics of a component.
Remarks If the object has an open designer, the Designer property returns the open designer; otherwise a new designer is created. The designer is a characteristic of certain VBComponent objects. For example, when you create certain types of VBComponent object, a designer is created along with the object. A component

can have only one designer, and it's always the same designer. The **Designer** property enables you to access a component-specific object. In some cases, such as in <u>standard modules</u> and <u>class modules</u>, a designer isn't created because that type of **VBComponent** object doesn't support a designer.

The **Designer** property returns **Nothing** if the **VBComponent** object doesn't have a designer.





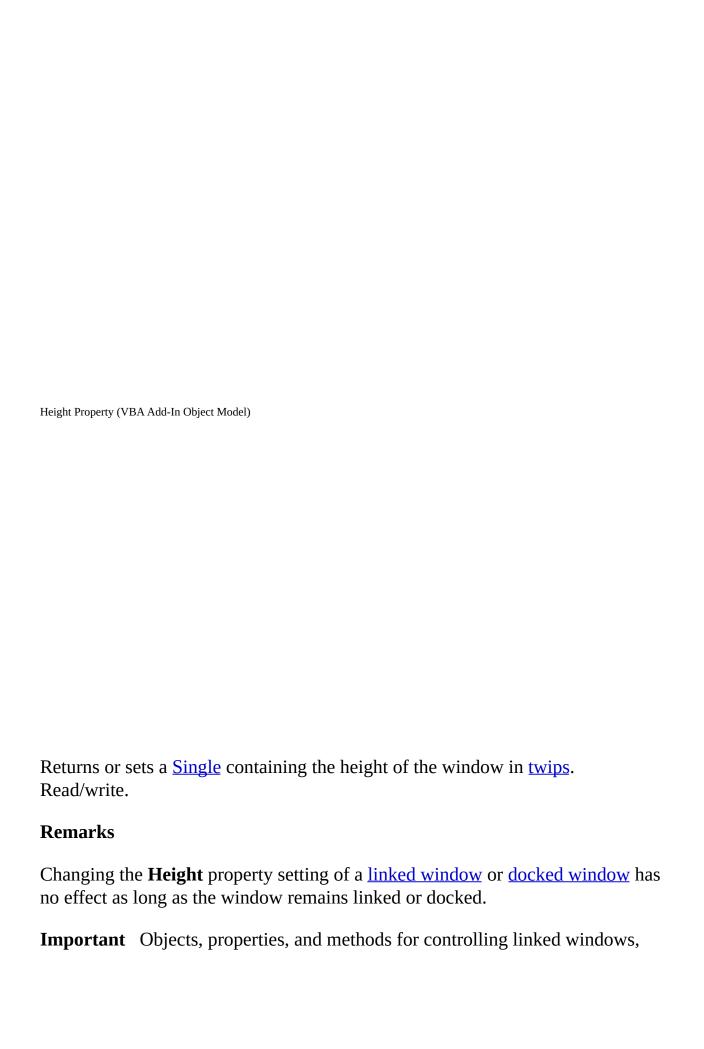
HasOpenDesigner Property

Returns a <u>Boolean</u> value indicating whether or not the **VBComponent** object has an open <u>designer</u>. Read-only.

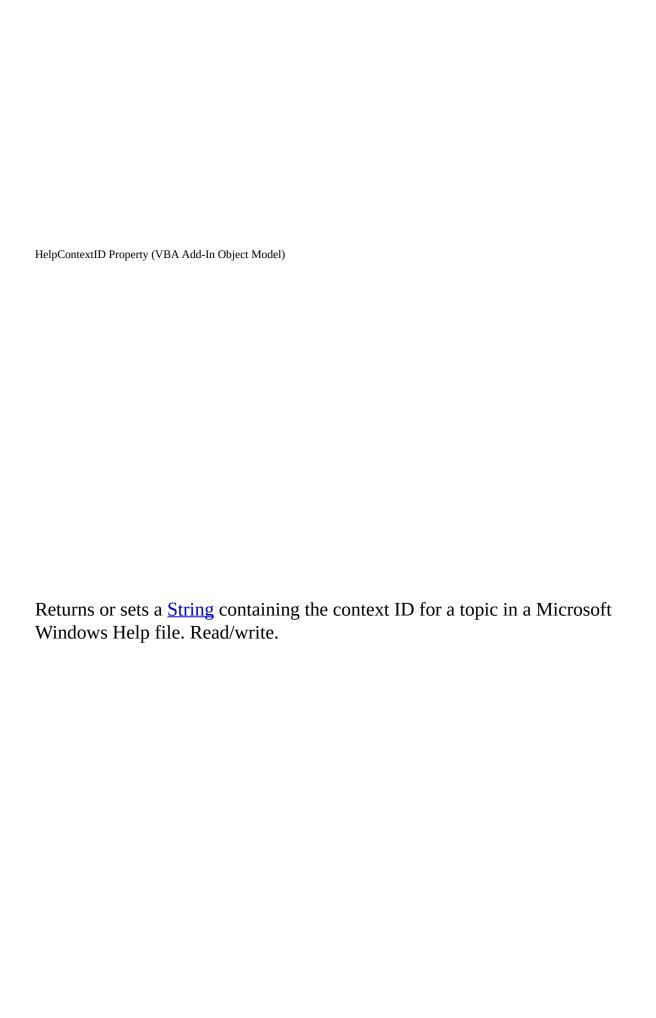
Return Values

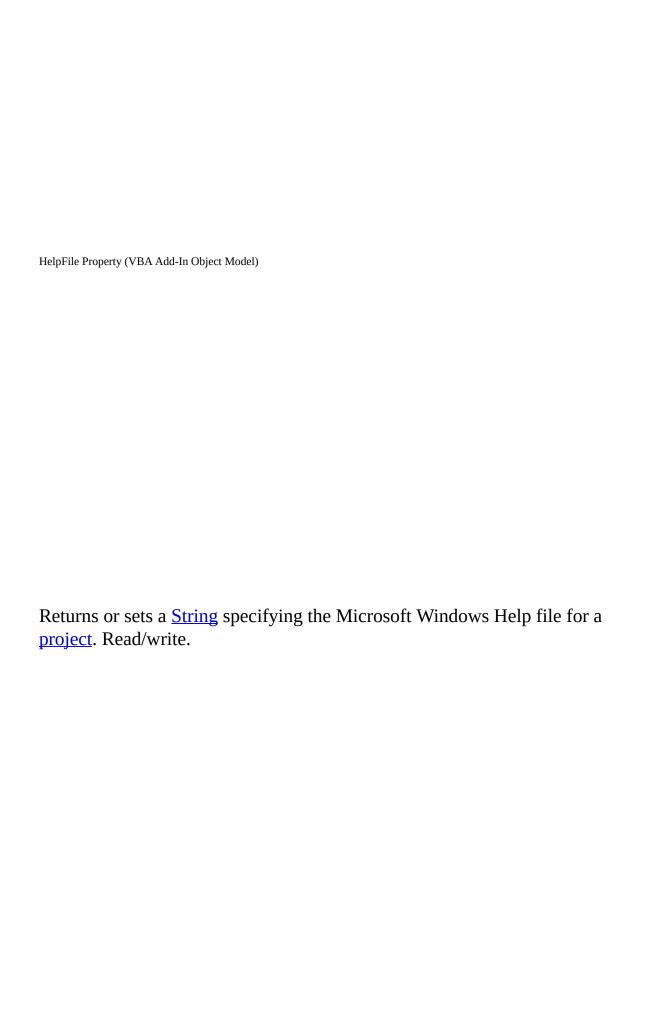
The **HasOpenDesigner** property returns these values:

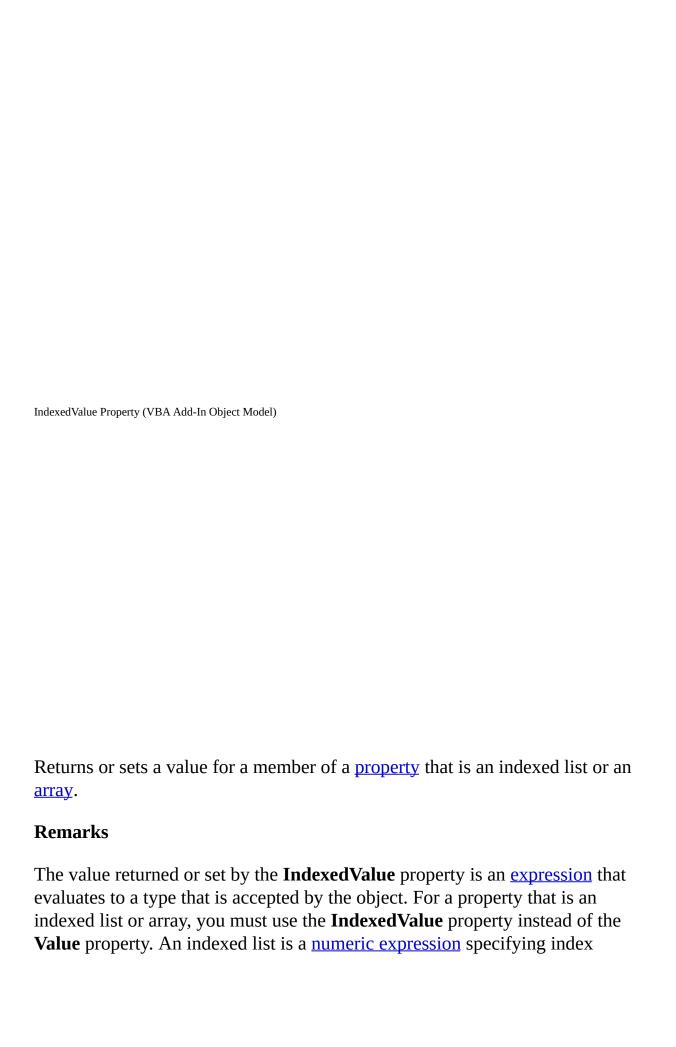
Value	Description
True	The VBComponent object has an open Design window.
False	The VBComponent object doesn't have an open Design window.



linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.







position.

IndexedValue accepts up to 4 indices. The number of indices accepted by **IndexedValue** is the value returned by the **NumIndices** property.

The **IndexedValue** property is used only if the value of the **NumIndices** property is greater than zero. Values in indexed lists are set or returned with a single index.

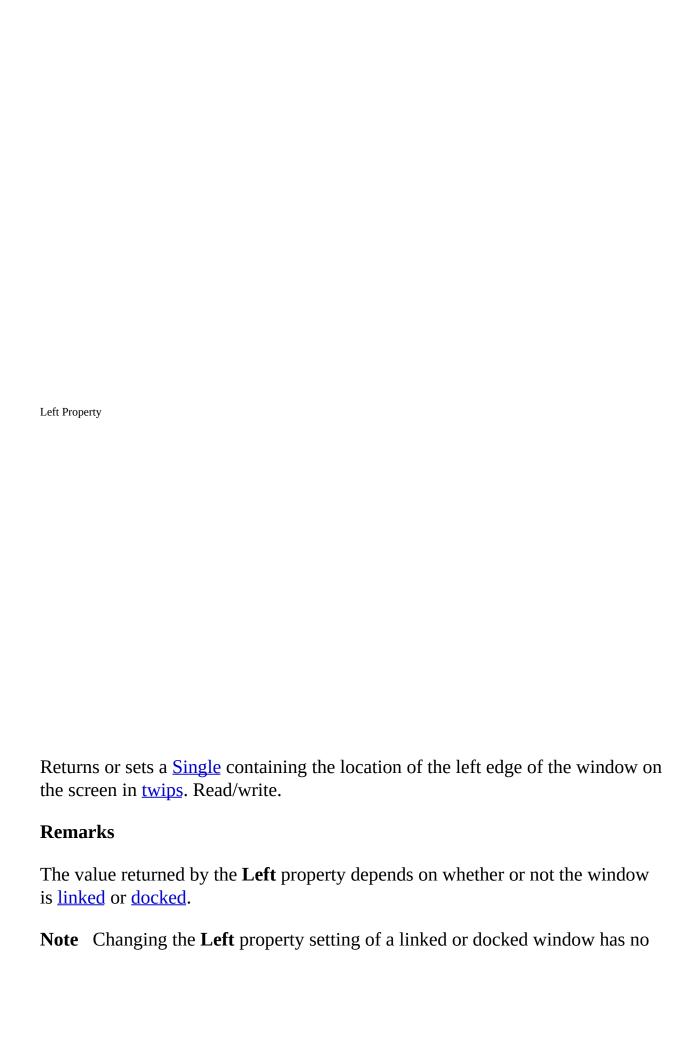
IsBroken Property

Returns a <u>Boolean</u> value indicating whether or not the **Reference** object points to a valid reference in the <u>registry</u>. Read-only.

Return Values

The **IsBroken** property returns these values:

Value	Description
True	The Reference object no longer points to a valid reference in the registry.
False	The Reference object points to a valid reference in the registry.



effect as long as the window remains linked or docked.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

Lines Property	
Data	
	containing the specified number of lines of code.
Syntax	
	artline, count) As String
The Lines syntax has these parts:	
Part	Description

object Required. An object expression that evaluates to an object in

the Applies To list.

startline Required. A Long specifying the line number in which to start.

count Required. A **Long** specifying the number of lines you want to

return.

Remarks

The <u>line numbers</u> in a <u>code module</u> begin at 1.

LinkedWindowFrame Property
Returns the Window object representing the frame that contains the window. Read-only. Remarks
The LinkedWindowFrame property enables you to access the object representing the <u>linked window frame</u> , which has properties distinct from the window or windows it contains. If the window isn't linked, the LinkedWindowFrame property returns Nothing .
Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

MainWindow Property
Returns a Window object representing the main window of the Visual Basic development environment. Read-only.
Remarks
You can use the Window object returned by the MainWindow property to add or remove <u>docked windows</u> . You can also use the Window object returned by the MainWindow property to maximize, minimize, hide, or restore the main window of the Visual Basic development environment.
Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

Major Property
Returns a Long containing the major version number of the referenced type library . Read-only.
Remarks
The number returned by the Major property corresponds to the major version number stored in the type library to which you have set the reference.

Minor Property
Returns a <u>Long</u> indicating the minor version number of the referenced <u>type</u> <u>library</u> . Read-only.
Remarks
The number returned by the Minor property corresponds to the minor version number stored in the type library to which you have set the reference.

Returns a value containing the mode of the specified <u>project</u>. Read-only.

Return Values

The **Mode** property return values are:

Constant	Description
vbext_vm_Run	The specified project is in run mode.
vbext_vm_Break	The specified project is in break mode.
vbext_vm_Design	The specified project is in design mode.



Property object and the **Reference** object, read-only.

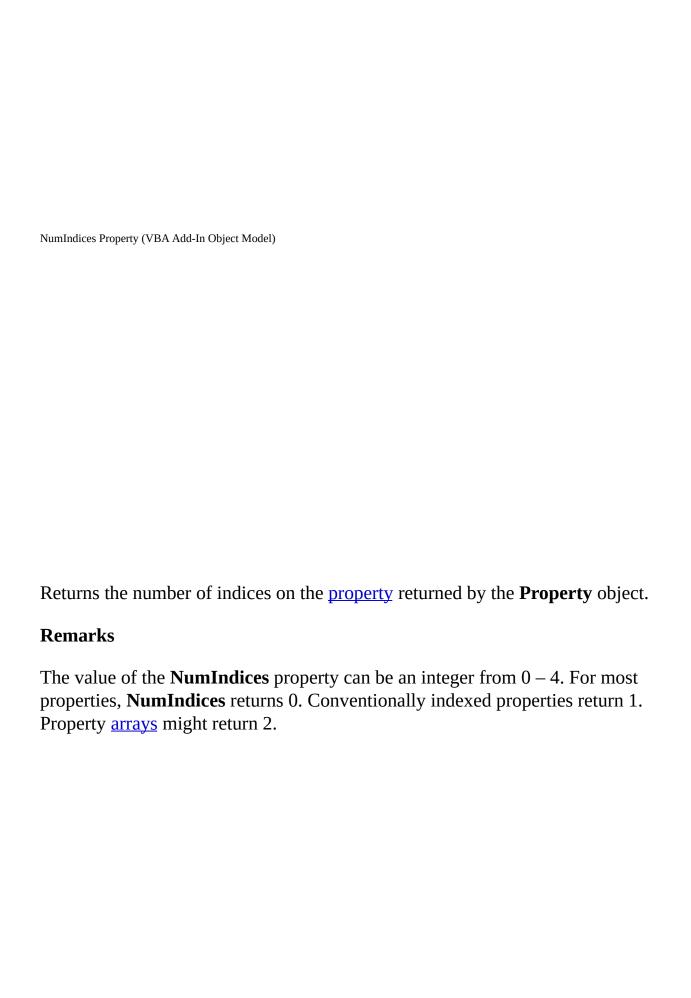
Remarks

The following table describes how the **Name** property setting applies to different objects.

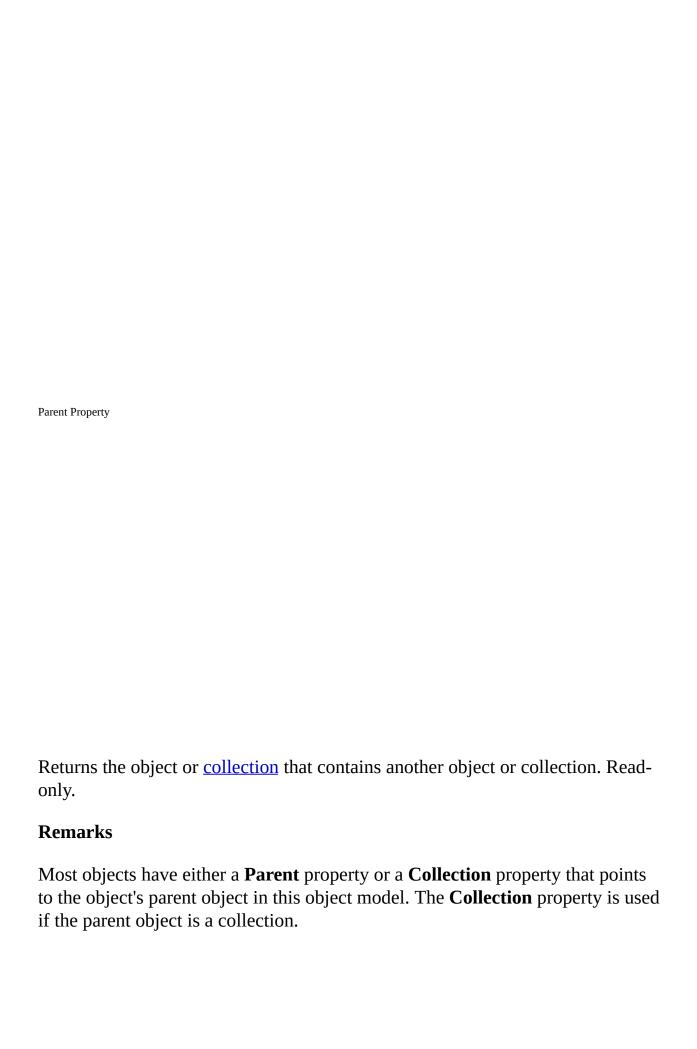
Object	Result of Using Name Property Setting
VBProject	Returns or sets the name of the active project.
VBComponent	Returns or sets the name of the component. An error occurs if you try to set the Name property to a name already being used or an invalid name.
Property	Returns the name of the property as it appears in the Property Browser . This is the value used to index the Properties <u>collection</u> . The name can't be set.
Reference	Returns the name of the reference in code. The name can't be set.

The default name for new objects is the type of object plus a unique integer. For example, the first new Form object is Form1, a new Form object is Form1, and the third TextBox control you create on a form is TextBox3.

An object's **Name** property must start with a letter and can be a maximum of 40 characters. It can include numbers and underline (_) characters but can't include punctuation or spaces. Forms and modules can't have the same name as another public object such as **Clipboard**, **Screen**, or **App**. Although the **Name** property setting can be a keyword, property name, or the name of another object, this can create conflicts in your code.



Object Property
Returns or sets the value of an object returned by a property. Read/write.
Remarks
If a property returns an object, you must use the Object property to return or set the value of that object.



Use the **Parent** property to access the <u>properties</u>, <u>methods</u>, and <u>controls</u> of an object's parent object.

Protection Property
Returns a value indicating the state of protection of a <u>project</u> . Read-only.

The specified project isn't protected.

Return Values

vbext_pp_none

Constant

The **Protection** property return values are:

Description

vbext_pp_locked The specified project is locked.

ReferencesEvents Property Returns the **ReferencesEvents** object. Read-only. **Settings** The setting for the <u>argument</u> you pass to the **ReferencesEvents** property is: Argument **Description** vbproject If *vbproject* points to **Nothing**, the object that is returned will supply events for the **References** collections of all **VBProject** objects in the **VBProjects** collection. If *vbproject* points to a valid **VBProject** object, the object that is returned will supply events for only the **References** collection for that <u>project</u>.

Remarks

The **ReferencesEvents** property takes an argument and returns an <u>event source</u> <u>object</u>. The **ReferencesEvents** object is the source for events that are triggered when references are added or removed.

Returns a <u>Boolean</u> value indicating whether or not the object was edited since the last time it was saved. Read/write.

Return Values

The **Saved** property returns these values:

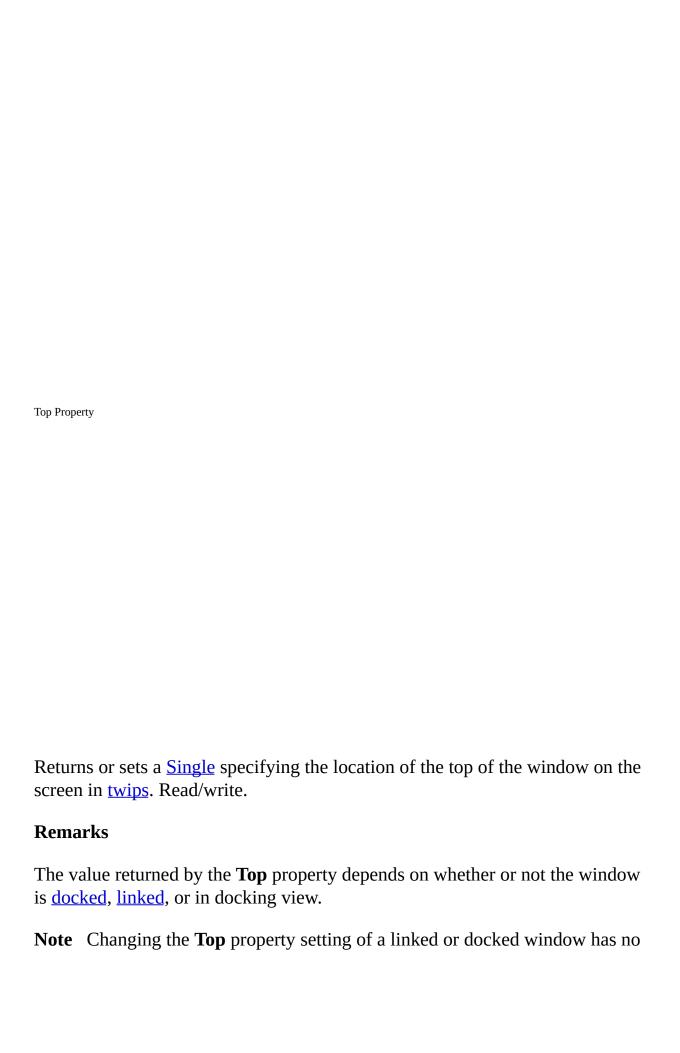
Value	Description
True	The object has not been edited since the last time it was saved.
False	The object has been edited since the last time it was saved.

Remarks

The **SaveAs** method sets the **SaveA** property to **True**.

Note If you set the **Saved** property to **False** in code, it returns **False**, and the object is marked as if it were edited since the last time it was saved.

SelectedVBComponent Property
Returns the selected component. Read-only.
Remarks
The SelectedVBComponent property returns the selected component in the Project window . If the selected item in the Project window isn't a component, SelectedVBComponent returns Nothing .



effect as long as the window remains linked or docked.

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

То	pLine Property
	eturns a Long specifying the line number of the line at the top of the <u>code pane</u> r sets the line showing at the top of the code pane. Read/write.
R	temarks
p	Use the TopLine property to return or set the line showing at the top of the code ane. For example, if you want line 25 to be the first line showing in a code ane, set the TopLine property to 25.
p	The TopLine property setting must be a positive number. If the TopLine roperty setting is greater than the actual number of lines in the code pane, the etting will be the last line in the code pane.



Returns a numeric or string value containing the type of object. Read-only.

Return Values

The **Type** property settings for the **Window** object are described in the following table:

Constant	Value	Description
vbext_wt_CodeWindow	0	Code window
vbext_wt_Designer	1	<u>Designer</u>
vbext_wt_Browser	2	Object Browser
vbext_wt_Immediate	5	Immediate window
vbext_wt_ProjectWindow	6	Project window
vbext_wt_PropertyWindow	7	Properties window
vbext_wt_Find	8	Find dialog box
vbext_wt_FindReplace	9	Search and Replace dialog box
vbext_wt_LinkedWindowFrame	e 11	Linked window frame
vbext_wt_MainWindow	12	Main window
vbext_wt_Watch	3	Watch window
vbext_wt_Locals	4	Locals window
vbext_wt_Toolbox	10	Toolbox
vbext_wt_ToolWindow	15	Tool window

Important Objects, properties, and methods for controlling linked windows, linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.

The **Type** property settings for the **VBComponent** object are described in the following table:

Constant	Value	Description
vbext_ct_StdModule	1	Standard module
vbext_ct_ClassModule	2	Class module
vbext_ct_MSForm	3	Microsoft Form
vbext_ct_ActiveXDesigner	11	ActiveX Designer
vbext_ct_Document	100	Document Module

The **Type** property settings for the **Reference** object are described in the following table:

Constant	Value	Description
vbext_rk_TypeLib	0	Type library
vbext_rk_Project	1	<u>Project</u>

The **Type** property settings for the **VBProject** object are described in the following table:

Constant	Value	Description
vbext_pt_HostProject	100	Host Project
vbext_pt_StandAlone	101	Standalone Project

Value Property
Returns or sets a <u>Variant</u> specifying the value of the <u>property</u> . Read/write.
Remarks
Because the Value property returns a Variant , you can access any property. To access a list, use the IndexedValue property.
If the property that the Property object represents is read/write, the Value property is read/write. If the property is read-only, attempting to set the Value property causes an error. If the property is write-only, attempting to return the Value property causes an error.
The Value property is the default property for the Property object.

VBE	Property
Re	turns the root of the VBE object. Read-only.
Re	marks
All	objects have a VBE property that points to the root of the VBE obj

Version Property
Returns a <u>String</u> containing the version of Visual Basic for Applications that the application is using. Read-only.
Remarks
The Version property value is a string beginning with one or two digits, a period, and two digits; the rest of the string is undefined and may contain text or numbers.

Property

For the **Window** object, returns or sets a <u>Boolean</u> value that specifies the visibility of a window. Read/write. For the **CodePane** object, returns a **Boolean** value that indicates whether or not the <u>code pane</u> is visible in the window. Read-only.

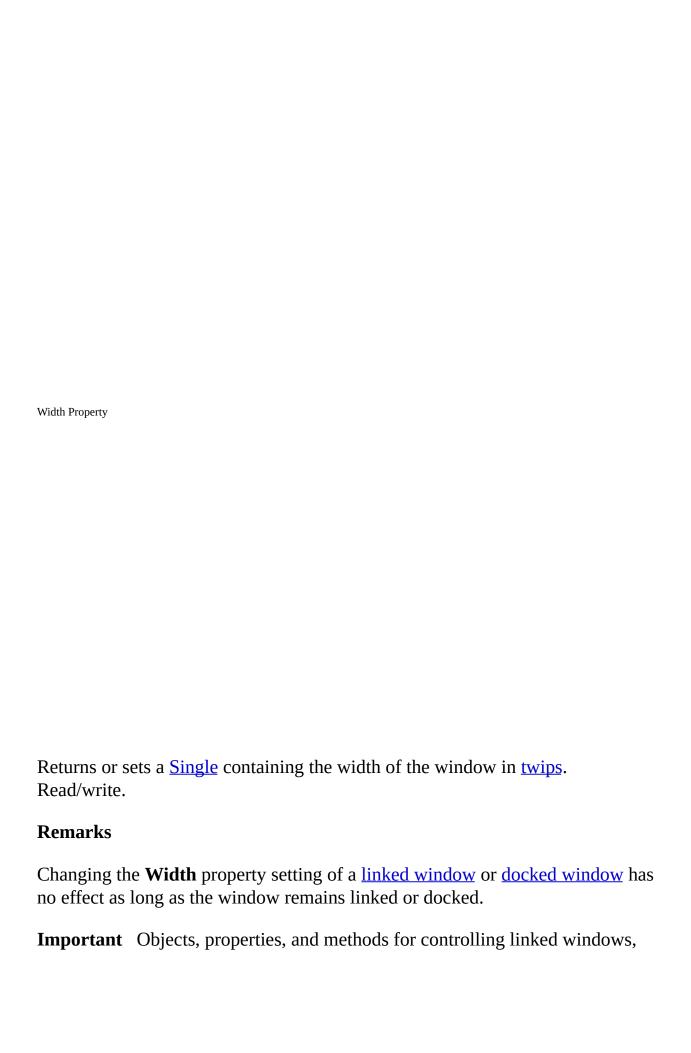
Return Values

The **Visible** property returns the following values:

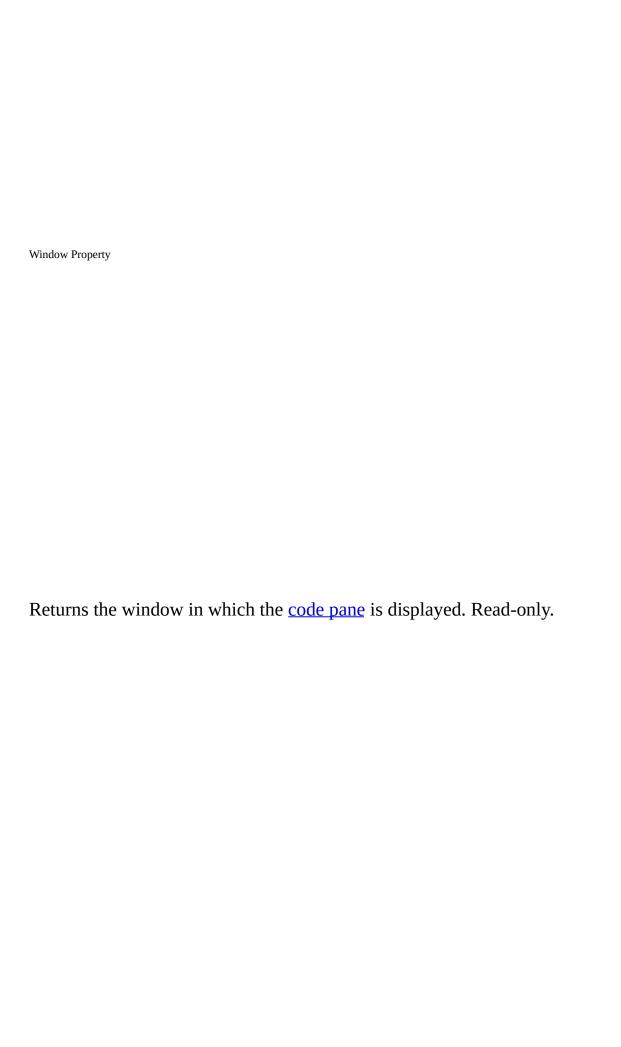
Value Description

True (Default) Object is visible.

False Object is hidden.



linked window frames, and docked windows are included on the Macintosh for compatibility with code written in Windows. However, these language elements will generate run-time errors when run on the Macintosh.



WindowState Property

Returns or sets a numeric value specifying the visual state of the window. Read/write.

Settings

The **WindowState** property returns or sets the following values:

Constant	Value	Description
vbext_ws_Normal	0	(Default) Normal
vbext_ws_Minimize	1	Minimized (minimized to an icon)
vbext_ws_Maximize	2	Maximized (enlarged to maximum size)

AddIns Collection



Returns a collection of add-ins registered for VBA.

Syntax

object.AddIns

Remarks

The **AddIns** collection is accessed through the **VBE** object. Every add-in listed in the Add-In Manager in VBE.has an object in the **AddIns** collection.