Microsoft Internet Explorer Scripting Object Model

The Microsoft® Internet Explorer scripting object model is a structure for embedding VBScript and JScript in HTML documents. The Internet Explorer model is compatible with Netscape Navigator™ with the object model used in the JScript™ language.

The Microsoft Internet Explorer object model is accessible from any scripting language that is compatible with the ActiveX scripting framework, such as Microsoft Visual Basic® Scripting Edition (VBScript). This document provides an overview of the object model, sample code (in VBScript and in JScript), and reference information. This document includes descriptions of the methods, properties, and events used with scripting engines in Internet Explorer.

Notes

1 All properties and methods that modify the HTML contents must be called during HTML parse time. The code must reside in a script block that runs inline during the loading of the HTML document. This is called *immediate execution* in the ActiveX Scripting Model.

2 VBScript uses parentheses, "( )", for indexing arrays; JScript uses square brackets, "[ ]", and both use "( )" for passing function and method arguments, when appropriate. Most examples in this document are in VBScript.

3 JScript is case sensitive; VBScript is not.

Attaching and Invoking Scripts

There are three ways to attach and invoke scripts in HTML:

- Use the SCRIPT tag.
- Use those attributes of HTML elements that support scripts.
- Use a custom URL type. (Note: This method does not apply for VBScript.)

Using the SCRIPT Element

Use the SCRIPT element to add scripts to HTML. SCRIPT is an element for embedding script code in a document. Using SCRIPT, the full source code of a script can be included within the document. The SCRIPT element can be used to point to external scripts, using the SRC attribute.

For example, this HTML describes a page with a SCRIPT element which includes code written in VBScript:

```
<SCRIPT language="VBScript">
  document.write("Hello, Webmaster.")
</SCRIPT>
```

The example in JScript would read:

```
<SCRIPT language="Javascript">
    //... Additional JScript statements ...
</SCRIPT>
```

Evaluation of SCRIPT

The SCRIPT element is evaluated when the document is loaded. All code is executed at load time in the order in which it appears in the document. Therefore, any reference to an object must appear in the text *after* the script element in which the object is defined.

The document object's write method can insert both text and objects—such as buttons and ActiveX controls. These objects can be referenced only in a script block following the script block that defined them. You will be able to refer to and copy references to objects that are the result of a code download. You can invoke any method on an object—but only when the object has been downloaded.

Using Scripts as Attributes of HTML Elements

Another way to insert scripts is to use the attributes of HTML elements that support scripts. When these attributes match with events on the elements, the script is executed when the event occurs. This can be done with HTML elements, such as forms, buttons, or links; however, this method does not work for items inserted using the OBJECT tag.

The following example uses this syntax in Button1 to handle the onClick event. To demonstrate the ability to combine multiple scripting languages on the same page, the scriptlet for Button1 is implemented in VBScript and that for Button2 in JScript.

```
<form name="Form1">
  <input type="button" name="Button1" value="VB"
    onClick="pressed" language="VBScript">
  <input type="button" name="Button2" value="Java"
    onClick="pressed2( )" language="Javascript">
</form>

<SCRIPT language="VBScript">
  sub pressed
    document.Form1.Button1.value="VBScript"
    alert "Pressed the VBScript button"
  end sub
</script>
<script language="Javascript">
  function pressed2( )
  {
    document.Form1.Button2.value="Javascript"
    alert("Pressed the Java button.")
  }
</script>
```

Note the use of the language attribute on the input tag to indicate the scriptlet's language. If no language is specified, the scriptlet defaults to the language of the most recently encountered script block. If no script block has been encountered, the language defaults to JScript.

The elements FORM, INPUT, BODY, and A support this syntax, but with differing events. See the individual tags referenced later in this document.

An Alternative Using SCRIPT

The SCRIPT element can also be used with the FOR="object" EVENT="eventname" syntax. This method can be used for any named elements, and for any elements inserted using the OBJECT tag. The following example is similar to the previous script example, but it uses a different syntax:

```
<form name="Form1">
  <input type="button" name="Button1" value="Click">
  <script for="Button1" event="onClick" language="VBScript">
    alert "Button has been pressed"
    document.Form1.Button1.value="PRESSED"
  </script>
</form>
```

Using Scripts in URLs

Scripts can be invoked using the A element combined with a custom URL type. This allows a script to be executed when the user clicks on a hyperlink. This URL type is valid in any context, but is most useful when used with the A element. For example:

```
<A HREF="javascript:alert('hi there')">Click me to see a message.</A>
```

Syntax

script-engine:script-code

Executes the script code using the script engine when the URL is resolved. For example, to execute a script when the user clicks on a hyperlink, use:

```
<title> JScript example </title>
<A HREF="javascript:alert(document.title)">Click here to see the title of
the current document.</A>
```

Notice that the script is executed in the context of the current page, which means that document.title evaluates to the document containing the script.

Argument	Type	Description
script-engine	String	A string that names a scripting engine (*must* be JScript for Beta 1).
script-code	String	A string that evaluates to a script in the syntax supported by the scripting engine. This script is executed by the scripting engine when the URL is evaluated.

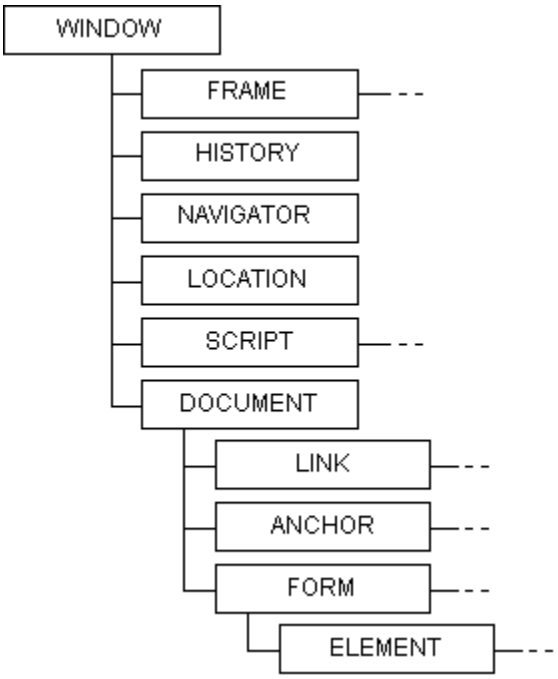
Note This syntax is only supported for JScript; in particular, VBScript: will not work in Internet Explorer. Also, the JScript: syntax is currently supported only from scripts, not when typed into the address bar by users.

Object Hierarchy and Scoping

There are eleven objects to consider in the HTML object model:

- Window
- **Frame**
- History
- Navigator
- Location
- **Script**
- Document
- Link
- Anchor
- Form
- Element

These objects are organized in the following hierarchy (the dotted line following an object indicates that multiple objects may exist):



Each of these objects has its own rules for scoping and containment.

The Window Object

The top level object is a window. Every window contains:

- **Frame** - Array of contained frame windows. Each frame is a window that has its own properties, including a document.
- History - History object for the current window. This object is used to access the history list from the browser.
- Navigator - Navigator object for the current window. The navigator object contains information about the browser application.
- Location - Location object for the current window. Provides information about the location of the window's URL.
- **Script** - Scripting function defined using the SCRIPT element in the window scope.
- Document - Document in the current window.

The window object properties can be referenced directly by scripts while in the window scope. So, for example, script authors do not need to type:

```
window.name
```

to reference the window name; instead, it is sufficient just to type:

```
name
```

Note also that it is possible to call scripts from one window object to another. So, to execute the script myscript in the topmost window, use:

```
top.myscript( )
```

The Document Object

The document object is one level below the window object. This object contains:

- Link - Array of hyperlinks found on the given document.
- Anchor - Array of forms found on the given document.
- Form - Array of anchors found on the given document.

Because scripts attach to the window object, not the document object, the script author must type **document.***property* to access document properties. For example, to get the title of the document:

```
<script language="VBScript">
'...
string1 = document.title     -put the document title into string1
'...
</script>
```

To access the forms in a document, the author can either refer by name or through the form array. So, for the following form:

```
<form name="Form1">
     <input type="button" name="Button1" value="Press ME" onClick="pressed">
</form>
```

the author can access the object named button1 by name:

```
<script language="VBScript">
sub pressed
   document.Form1.Button1.value="I've been pressed"  '   access the form by
name
end sub
</script>
```

by index:

```
<script language="VBScript">
sub pressed
     document.forms(0).Button1.value="I've been pressed"   '   access the
form by index
end sub
</script>
```

or by index name and array reference.

Scripts can refer to contained elements that are not form types directly, without using **document**. So, for example, if the authors create an object called myObject, they can reference it directly in script as follows:

```
<object name="myObject" ... >
</object>

<script language="VBScript">
sub foo
     myObject.color = "green"     - access the form by index
end sub
</script>
```

The Form Object

The form object contains:

- Element - Array of objects and intrinsic controls contained in the form.

A script can reside either in a form or in a window. If a script lives outside the form, it needs to access the elements in the form, either by name or through the form array (see the example in "The Document Object"). If, however, the script element is inside the form, it can access the elements in the form directly.

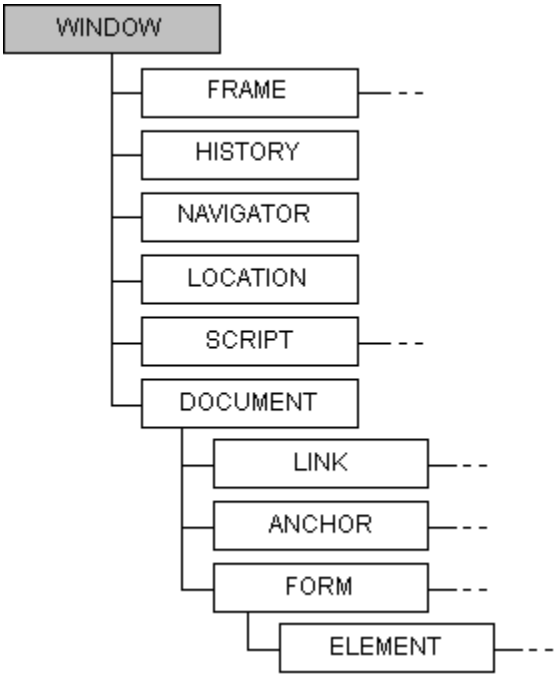
```
<form name="Form1">
     <input type="button" name="Button1" value="Press me">
     <script for="Button1" event="onClick" language="VBScript">
    alert "I've been pressed"
    document.Form1.Button1.value="Click"
    Button1.value="Click"
     </script>
</form>

<script language="VBScript">
sub foo
     document.Form1.Button1.value="Click"
end sub
</script>
```

window Object

The top level object in the scripting object model is a window. Every window contains:

- **Frame** - Array of frame windows contained by a parent window. Each frame is a window that has its own properties, including a document.
- History - History object for the current window. This object is used to access the history list from the browser.
- Navigator - Navigator object for the current window. The navigator object contains information about the browser application.
- **Location** - Location object for the current window. Provides information about the location of the window's URL.
- **Script** - Scripting function defined using the SCRIPT element in the window scope.
- Document - Document in the current window.



The window object represents the Internet Explorer window and its methods and properties. Methods and properties of the window object can be called by scripts directly. This means that if you wanted to get the name of the current page, you would use the following script (Notice that the property name does not need a prefix):

```
<script language="VBScript">
     '...
     string1=name          - get the name of the current window
     alert string1          - display that name as an alert
     '...
</script>
```

However, you can access the properties of other window objects without explicitly mentioning the window. For example, to get the name of the current window's parent, you would use:

```
<script language="VBScript">
     '...
     string1=parent.name       - get the name of the parent window
     '...
</script>
```

Window events can be hooked to scripts using extensions to the BODY tag. To add scripts to a window event, add a script for either the onLoad or onUnload events in the BODY tag at the top of the page. In the following example, the *Foo* function is called when the page is loaded:

```
<HTML>
...
<BODY Language="VBScript" onLoad="Foo">
...
<SCRIPT language="VBScript">
...
Sub Foo
  MsgBox "This is sub foo"
End Sub
...
</SCRIPT>
....
</BODY></HTML>
```

To access a window by name, the window must be given a name. This can happen in three ways: by using the *window.***open** method, by creating the window with a name using the FRAMESET element, or by creating the window with a URL using the TARGET attribute.

The following examples all create a window named *foo* with contents *a.htm*.

```
<SCRIPT Language="VBScript">
window.open ("a.htm", "foo");
</SCRIPT>

<FRAMESET cols = "200, *" frameborder=0>
  <FRAME name = "foo" src="a.htm">
  <FRAME name = "bar" src="b.htm">
</FRAMESET>

<A HREF="a.htm" TARGET = "foo">Click here to see a.htm in window foo.</A>
```

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

Properties

Window properties can be referenced directly in the scripting language. Consequently, all window properties are reserved words and cannot be used as variable names in procedures. The following window properties are used:

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

name Property

Returns the name of the current window. This property is read-only.

Syntax

[*window.*]**name**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns the string containing the current window name or "null" if none.

Remarks

To set the value of String1 to be the name of the current window, use:

```
String1=name.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

parent Property

Returns the window object of the window's parent. This property is read-only. The parent of the window is the containing frame. If the current window has no containing frame windows, then the parent evaluates to the current window.

Syntax

[*window.*]**parent**

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns the window object that evaluates to the parent window.

Remarks

To set the value of String1 to be the name of the parent of the current window, use:

```
String1=parent.name.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

opener Property

Returns the window object of the window that opened the current window.

Syntax

[*window.*]**opener**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns the window object that evaluates to the opener window.

Remarks

To set the value of "String1" to be the opener of the current window, use:

```
String1=opener.name.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, self, top, location, defaultStatus, status, frames, history, navigator, document

self Property

Returns the window object of the current window. This property is read-only.

Syntax

[*window.*]**self**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object that evaluates to the current window.

Remarks

To set the value of String1 to be the name of the current window, use:

```
String1=self.name
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, top, location, defaultStatus, status, frames, history, navigator, document

top Property

Returns the window object of the topmost window. This property is read-only. The topmost window is the containing window of all frames in the current browser instance.

Syntax

[*window.*]**top**

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object that evaluates to the topmost window.

Remarks

To set the value of String1 to be the name of the topmost window, use:

```
String1=top.name.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, location, defaultStatus, status, frames, history, navigator, document

location Property

Returns the location object for the current window. For more details, see "location Object."

Syntax

[*window.*]**location**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object that evaluates to the location object of window.

Remarks

To set the value of String1 to be the name of the URL of the current window, use:

```
String1=location.HRef.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, defaultStatus, status, frames, history, navigator, document

defaultStatus Property

Sets the default status text in the lower left portion of the status bar.

Syntax

[*window.*]**defaultStatus**[**=***string*]

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

string
Optional. Sets the default status text to the value of string.

Remarks

To set the default status to "Hello" use:

```
defaultStatus="Hello"
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, status, frames, history, navigator, document

status Property

Sets the status text in the lower left part of the status bar.

Syntax

[*window.*]**status**[**=***string*]

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

string
Optional. Sets the status text to the value of string.

Remarks

To set the status to "Hello" use:

```
status="Hello"
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, frames, history, navigator, document

frames Property

Returns the array of frames for the current window.

Syntax

[*window.*]**frames**[*integer*]

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object expression that evaluates to the array of frames.

Remarks

To set String1 to the URL of frame[0], use:

```
String1=parent.frames[0].location.href.
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, history, navigator, document

history Property

Returns the history object of the current window. For more details on methods, properties, and events, see "history Object."

Syntax

[*window.*]**history**

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object expression that evaluates to a history object.

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, navigator, document

navigator Property

Returns the navigator object of the current window. For more details on methods, properties, and events, see "navigator Object."

Syntax

[*window.*]**navigator**

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object expression that evaluates to a navigator object.

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, document

document Property

Returns the document object of the current window. For more details on methods, properties, and events, see "document Object."

Syntax

[*window.*]**document**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object expression that evaluates to a document object.

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator

Methods

This section describes the methods for the window object.

alert Method

Displays an alert message box.

Syntax

[*window.*]**alert (***string***)**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

string

A string containing the text to display in the message box.

Remarks

The following example would display an alert that contained the string "Hello World":

```
alert ("Hello World")
```

Applies To

Window

Methods

confirm, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

confirm Method

Displays a message box that allows the user to select **OK** or **Cancel**.

Syntax

[*bool* **=**][*window.*]**confirm (***string***)**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

string

A string containing the text to display in the message box.

Return Values

Returns the user response: TRUE if the user pressed OK; FALSE if not.

Remarks

The following example would display a message box that contained the string "Do you want to continue?":

```
x=confirm ("Do you want to continue?")
```

Applies To

Window

Methods

alert, prompt, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

prompt Method

Prompts the user for input.

Syntax

[*string* **=**][*window.*]**prompt** [*prompt*] [**,** *default*]

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

prompt
Optional. A string containing the text to display in the message box.

default
Optional. A string containing the default text to display in the input field.

Return Values

Returns the user input.

Applies To

Window

Methods

alert, confirm, open, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

open Method

Creates a new window.

Syntax

[*newwindow* **=**][*window.*]**open** *url*, *target*, ["[**toolbar=***bool*] [, **location=***bool*][, **directories=***bool*][, **status=***bool*][, **menubar=***bool*][, **scrollbars=***bool*][, **resizeable=***bool*][, **width=***pixels*][, **height=***pixels*]"] [, **top=***pixels*][, **left=***pixels*]

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

url
A string containing a correctly parsed URL. The URL is parsed identically to a link—both relative and absolute paths are supported.

target
A string containing the name of the target window. If a window with this name already exists, the existing window is reused with the new URL. If the window does not exist, a new window is created with that name. Note that this works identically to the TARGET attribute of an HREF in HTML.

bool
The remaining window properties are passed as a comma-separated list. Most of these can be set to Boolean values, either *yes* or *no* [*1* or *0*]. These properties are toolbar, location, directories, status, menubar, scrollbars, and resizeable.

pixels
Four other properties in this list, top, left, width, and height, have values in pixels.

Return Values

Returns an object expression that evaluates to the created window object.

Remarks

The following example would create a new window:

```
 open "http://www.microsoft.com", "myWindow", "toolbar=no, location=no,
directories=no"
```

Applies To

Window

Methods

alert, confirm, prompt, close, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

close Method

Closes the window.

Syntax

[*window.*]**close**

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

Return Values

Returns an object expression that evaluates to the indexed frame.

Applies To

Window

Methods

alert, confirm, prompt, open, setTimeout, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

setTimeout Method

Sets a timer to call a function after a specified number of milliseconds.

Syntax

ID = [*window.*]**setTimeout** *expression*, *msec*, *language*

Parts

window
Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

expression
An object expression that evaluates to a function or object property. This function is called when the Timeout is set.

msec
The number of milliseconds that passes before the expression is evaluated.

language
The scripting language used. Can be "VBScript" or "JScript".

Return Values

Returns the ID of the timer object. This can be used to cancel the timer using the **clearTimeout** method.

Remarks

To call Button1.Click after 100 milliseconds, use:

```
MyID = setTimeout ("Button1.Click", 100).
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, clearTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

clearTimeout Method

Clears the timer having a particular ID.

Syntax

[*window.*]**clearTimout** *ID*

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

ID

The ID of the timer to be cleared. If there is no timer with this ID, the function does nothing.

Remarks

To clear the timer with ID=MyID, use:

```
clearTimeout MyID
```

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, navigate

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

navigate Method

Navigates the window to a new URL.

Syntax

[*window.*]**navigate** *url*

Parts

window

Optional. An object expression that evaluates to a window object. If omitted, the current script window is used.

url

A string containing a valid URL. The URL can be either relative or absolute.

Applies To

Window

Methods

alert, confirm, prompt, open, close, setTimeout, clearTimeout

Events

onLoad, onUnload

Properties

name, parent, opener, self, top, location, defaultStatus, status, frames, history, navigator, document

onLoad Event Handler

Fired after all HTML has been parsed and processed.

Syntax

onLoad=*function-name*

Values

function-name
An object expression that evaluates to a scripting function.

Remarks

To call the VBScript function *Foo* when the page is loaded, use:

```
<BODY Language="VBScript" onLoad="Foo">
```

Applies To

Window

onUnload Event Handler

Fired when the contents of the window are unloaded.

Syntax

onUnload=*function-name*

Values

function-name
An object expression that evaluates to a scripting function.

Remarks

To call the VBScript function *Foo* when the page is unloaded, use:

```
<BODY Language="VBScript" onUnload="Foo">
```

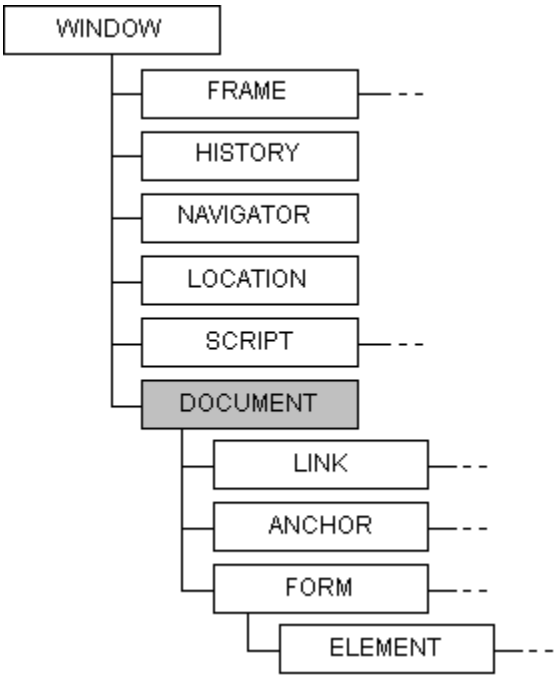
Applies To

Window

document Object

An object that resides below the window in the scripting object model. A document may contain:

- Link - Array of hyperlinks found on the given document.
- Anchor - Array of anchors found on the given document.
- Form - Array of forms found on the given document.



The document object reflects the HTML document currently in the browser and objects on the page—links, forms, buttons, and ActiveX objects. Methods and properties of the document object must be called in a script by placing *document* first in the statement. This means that if you wanted to set the background color on the page, the script would look like:

```
<script language="VBScript">
     document.bgColor = "Blue"
</script>
```

The document object currently has no events.

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

linkColor Property

Gets or sets the current color of the links in a document.

Syntax

document.**linkColor** [**=***rgb-value*|*string*]

Parts

document
An object expression that evaluates to a document object.
rgb-value
Optional. The new color of links in the document.
string
Optional. A string value specifying the color.

Return Values

Returns the rgb value of the current link color.

Remarks

Note that this property can only be set at parse time, not after the page is painted. So the code:

```
<SCRIPT LANGUAGE="JScript">
document.vLinkColor = "green";
document.linkColor = "red";
document.alinkColor = "aqua";
</SCRIPT>
```

sets the link color, while the code:

```
<FORM>
"document.linkColor='#000000'">
<INPUT TYPE="button" VALUE="Set Visited Link Color to White" onClick =
"document.vLinkColor='#AAAAAA'">
</FORM>
```

will have no effect when the button is clicked.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

aLinkColor Property

Gets or sets the current color of the *active* links in a document. A link is active when the pointer is positioned over it and the mouse button is pressed and not released. Note that Internet Explorer does not have this feature, so **aLinkColor** has no effect; however, it is supported in the object model for compatibility reasons. As with **linkColor**, this property can only be set at parse time. See the examples in **linkColor** for details.

Syntax

document.**aLinkColor** [**=***rgb-value*|*string*]

Parts

document
An object expression that evaluates to a document object.

rgb-value
Optional. The new color of links in the document.

string
Optional. A string value specifying the color.

Return Values

Returns the rgb value of the current link color.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

vLinkColor Property

Gets or sets the current color of the visited links in a document. As with **linkColor**, this property can only be set at parse time. See the examples in **linkColor** for details.

Syntax

document.**vLinkColor** [**=***rgb-value*|*string*]

Parts

document
An object expression that evaluates to a document object.

rgb-value
Optional. The new color of links in the document.

string
Optional. A string value specifying the color.

Return Values

Returns the rgb value of the current link color.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

bgColor Property

Gets or sets the current color of the background in a document.

Syntax

document.**bgColor** [**=***rgb-value*|*string*]

Parts

document
An object expression that evaluates to a document object.
rgb-value
Optional. The new color of the background in the document.
string
Optional. A string value specifying the color.

Return Values

Returns the rgb value of the current background color.

Remarks

To set the background color to white, use:

```
document.bgColor="000000"
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

fgColor Property

Gets or sets the foreground color.

Syntax

document.**fgColor**[**=***rgb-value*]

Parts

document
An object expression that evaluates to a document object.

rgb-value
Optional. The new color of the foreground in the document.

Remarks

To set the foreground color to white, use:

```
document.fgColor="000000"
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

anchors Property

Returns the array of anchors in a document.

Syntax

document.**anchors**[*integer*]

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns an object expression that evaluates to the array of anchors.

Remarks

To access the first anchor in the document, use:

```
document.anchors[0]
```

To get the length of the anchors array, use:

```
document.anchors.length
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, links, forms, location, lastModified, title, cookie, referrer

links Property

Returns the array of links for the current document.

Syntax

document.**links** [*integer*]

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns an object expression that evaluates to the array of links.

Remarks

To access the first link in the document, use:

```
document.links[0]
```

To get the length of the links array, use:

```
document.links.length
```

Note that the locations in the links collection are read-only.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, forms, location, lastModified, title, cookie, referrer

forms Property

Returns the array of forms in a document.

Syntax

document.**forms** [*integer*]

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns an object expression that evaluates to the array of forms.

Remarks

To access the first form in the document, use:

```
document.forms[0]
```

To get the length of the forms array, use:

```
document.forms.length
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, location, lastModified, title, cookie, referrer

location Property

Returns a read-only representation of the location object.

Syntax

*document.***location**

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns an object expression that evaluates to the location object of the document.

Remarks

To set String1 to the document's URL, use:

```
String1 = document.location.href
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, lastModified, title, cookie, referrer

lastModified Property

Returns the last modified date of the current page.

Syntax

*document.***lastModified**

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns a string containing the date.

Remarks

To set Date1 to the document's last modified date, use:

```
Date1 = document.lastModified
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, title, cookie, referrer

title Property

Returns a read-only representation of the document's title.

Syntax

*document.***title**

Parts

document
An object expression that evaluates to a document object.

Return Values

Returns a string expression that evaluates to the title of the document.

Remarks

To set String1 to the document's title, use:

```
String1 = document.title
```

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, cookie, referrer

cookie Property

Gets or sets the cookie for the current document.

Syntax

document.**cookie** [**=***newcookie*]

Parts

document
An object expression that evaluates to a document object.

newcookie
Optional. The new value for the cookie. Because the cookie file is just a text file, this value is a string.

Return Values

Returns a string containing the current cookie.

Remarks

The cookie is a string expression stored for the current page. Note that setting the cookie overwrites any current cookie information. Also note that you can use string expressions to locate particular information in the cookie string.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, referrer

referrer Property

Gets the URL of the referring document.

Syntax

*document.***referrer**

Parts

document

An object expression that evaluates to a document object.

Return Values

Returns a string containing the URL of the referring document.

Currently returns the URL of the referring document when there is a referrer, and NULL when there is no referrer.

Remarks

The referring document is the document that contained the link the user clicked on to get to the current document. For example, if the user is on www.microsoft.com and clicks on a link to navigate to www.msn.com, the referrer property of the document for www.msn.com is www.microsoft.com. Note that by definition the referrer varies depending on how the user linked to the current document. If the user navigated to the document without clicking on a link from another page, referrer should return NULL.

Applies To

Document

Methods

write, writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie

write Method

Places the given string into the current document. Unless otherwise specified, the string is appended to the current document at the current position.

Syntax

*document.***write***string*

Parts

document
An object expression that evaluates to a document object.

string
The string to write to the current document. Note that the string is added into the HTML directly, so it must be formatted as HTML.

Remarks

The following examples demonstrate the use of the **write** method:

```
<HTML><BODY>
<SCRIPT LANGUAGE="VBScript">
document.write ("Hello world.")
</SCRIPT>
This is a document.
</BODY></HTML>
```

results in:

```
Hello world. This is a document.
```

Whereas:

```
<HTML><BODY>
This is a document.
<SCRIPT LANGUAGE="VBScript">
document.write ("Hello world.")
</SCRIPT>
</BODY></HTML>
```

results in:

```
This is a document. Hello world.
```

Applies To

Document

Methods

writeLn, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

writeLn Method

Places the given string into the current document with a new-line character appended to the end.

Syntax

*document.***writeLn** *string*

Parts

document
An object expression that evaluates to a document object.

string
The string to write to the current document. Note that the string is added into the HTML directly, so it must be formatted as HTML.

Remarks

This method is the same as the *document.***write** method with the addition of a newline character at the end. Note that a newline is ignored by HTML unless it is bracketed by <PRE> tags, so in many cases *document.***write** and *document.***writeLn** behave identically.

The following examples demonstrate the use of the **writeLn** method:

```
<SCRIPT LANGUAGE="VBScript">
document.writeLn ("Hello world.")
document.write ("Hello world.")
</SCRIPT>
```

results in:

```
Hello world.  Hello world.
```

Whereas:

```
<PRE>
<SCRIPT LANGUAGE="VBScript">
document.writeLn ("Hello world.")
document.write ("Hello world.")
</SCRIPT>
</PRE>
```

results in:

```
Hello world.
Hello world.
```

Applies To

Document

Methods

write, open, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

open Method

Opens the document stream for output.

Syntax

*document.***open**

Parts

document
An object expression that evaluates to a document object.

Remarks

Generally *document.***open** is followed by a sequence of *document.***write** or *document.***writeLn** statements, followed by *document.***close**. If the referenced document exists already, any information contained in the document is cleared. To write "Hello World" to the document, use:

```
document.open
document.writeLn "Hello World"
document.close
```

Note that this is identical to:

```
document.writeLn "Hello World"
```

with two exceptions.

- In the first example, "Hello World" is written to the screen after *document.***close**; in the second, it is written immediately.
- In the first example, *document.***open** clears the document if there is data; in the second, "Hello World" is appended to the end.

Applies To

Document

Methods

write, writeLn, close, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

close Method

Updates the screen to display all of the strings written after the last open method call.

Syntax

*document.***close**

Parts

document
An object expression that evaluates to a document object.

Applies To

Document

Methods

write, writeLn, open, clear

Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

clear Method

Closes the document output stream and writes the data to the screen. See the open method description for more information and examples.

Syntax

*document.***clear**

Parts

document
An object expression that evaluates to a document object.

Applies To

Document

Methods

write, writeLn, open, close

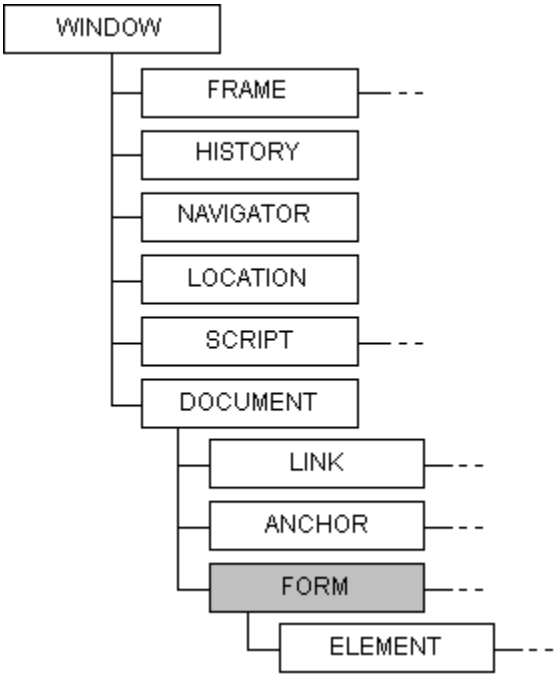
Properties

linkColor, aLinkColor, vLinkColor, bgColor, fgColor, anchors, links, forms, location, lastModified, title, cookie, referrer

form Object

An object that resides below the document in the scripting object model. A form may contain:

- Element - The array of objects and intrinsic controls contained in the form.



The form object represents a form in the HTML document. Forms are kept in the document object both as an array and by name. Script forms are accessible either by index (the documents forms array) or by name (given in the NAME="somename" attribute of the HTML <FORM> tag). Given a document with one form defined, the script can access the form in one of two ways:

```
<script language="VBScript">

' ...first method, by name ...
sub pressedByName
     document.Form1.Button1.value="I've been pressed"     ' access the form
by name
end sub

' ... second method, by index...
' Note that indexes start at 0, not 1!
sub pressedByIndex
     document.form1.elements(1).value="I've been pressed"     ' access the
form by index
end sub
</script>

<form name="Form1">
     <input type="button" name="Button1" value="Press ME"
onClick="pressedByName" language="VBScript">
     <input type="button" name="Button2" value="Press ME"
onClick="pressedByIndex" language="VBScript">
</form>
```

Methods

submit

Events

onSubmit

Properties

action, encoding, method, target, elements

action Property

Gets or sets the address to be used to carry out the action of the form.

Syntax

*form.***action**[**=***string*]

Parts

form
An object expression that evaluates to a form object.

string
Optional. A string containing the new action, generally a URL.

Return Values

Returns a string containing the current form action.

Remarks

If no URL is specified, the base URL of the document is used. Note that this is identical to changing the ACTION attribute of the <FORM> tag. So the script:

```
document.form[0].action = "http:// www.sample.com/bin/search"
```

is identical to the following:

```
<FORM ACTION="http:// www.sample.com/bin/search">
</FORM>
```

Applies To

Form

Methods

submit

Events

onSubmit

Properties

encoding, method, target, elements

encoding Property

Gets or sets the encoding for the form.

Syntax

*form.***encoding**[**=***string*]

Parts

form
An object expression that evaluates to a form object.

string
Optional. A string containing the new encoding. This must be a valid mime type, like "text/html."

Return Values

Returns a string containing the current form encoding.

Remarks

If no mime type is specified, "text/html" is used. Note that this is identical to changing the ENCTYPE attribute of the <FORM> tag. So the script:

```
document.form[0].action = "http:// www.sample.com/bin/search"
document.form[0].enctype = "text/html"
```

is identical to the following:

```
<FORM ACTION="http:// www.sample.com/bin/search" ENCTYPE="text/html">
</FORM>
```

Note that in the current build, encoding can be set, but has no effect on the operation of the form.

Applies To

Form

Methods

submit

Events

onSubmit

Properties

action, method, target, elements

method Property

Indicates how the form data should be sent to the server.

Syntax

*form.***method**[*string*]

Parts

form
An object expression that evaluates to a form object.

string
Optional. A string containing the new method, either GET or POST.

Return Values

Returns a string containing the current form method.

Remarks

GET means append the arguments to the action URL and open it as if it were an anchor; POST means send the data via an HTTP post transaction. Note that this is identical to the METHOD attribute of the <FORM> tag, so the script:

```
document.form[0].action = "http:// www.sample.com/bin/search"
document.form[0].method = "GET"
```

is identical to the following:

```
<FORM ACTION="http:// www.sample.com/bin/search" METHOD=GET>
</FORM>
```

Applies To

Form

Methods

submit

Events

onSubmit

Properties

action, encoding, target, elements

target Property

Specifies the name of the target window to display the form results in.

Syntax

*form.***target** [**=***string*]

Parts

form
An object expression that evaluates to a form object.

string
Optional. A string containing the new target name.

Return Values

Returns a string containing the current form target.

Remarks

Note that this is identical to the TARGET attribute of the <FORM> tag, so the script:

```
document.form[0].action = "http:// www.sample.com/bin/search"
document.form[0].target = "newWindow"
```

is identical to the following:

```
<FORM ACTION="http:// www.sample.com/bin/search" TARGET="newWindow">
</FORM>
```

Note that in the current build, **target** can be set; however, it has no effect on the operation of the form.

Applies To

Form

Methods

submit

Events

onSubmit

Properties

action, encoding, method, elements

elements Property

Returns the array of elements contained in the form.

Syntax

*form.***elements**[**=***string*]

Parts

form

An object expression that evaluates to a form object.

Return Values

Returns an object expression that evaluates to the array of elements in a form.

Remarks

The elements include any intrinsics (specified using the INPUT tag) or any embedded objects (specified using the OBJECT tag) contained in the form. So, the HTML:

```
<FORM ACTION="http:// www.sample.com/bin/search" METHOD=GET>
<INPUT NAME="aButton" TYPE ... >
<INPUT NAME="aCheckBox" TYPE ... >
<OBJECT NAME="anObject" DATA=...></OBJECT>
<INPUT NAME="aRadio" TYPE ... >
</FORM>
```

would generate an elements array where *form.***elements**.length returns 4, and *form.***elements**[2].name returns "anObject".

Applies To

Form

Methods

submit

Events

onSubmit

Properties

action, encoding, method, target

submit Method

Submits the form. Note that this is identical to clicking a form input with TYPE=SUBMIT.

Syntax

*form.***submit**

Parts

form

An object expression that evaluates to a form object.

Applies To

Form

Events

onSubmit

Properties

action, encoding, method, target, elements

onSubmit Event

Fired when the form is submitted.

Syntax

*form.***onSubmit =***action*

Values

form

An object expression that evaluates to a form object.

action

A string expression that evaluates to a scripting function call.

Remarks

This event can be used to prevent the form from being submitted, or it can be used to run additional code before the form is submitted. To prevent the form from being submitted, you must use "return <function>." So, the script:

```
form.onsubmit = "return IsValid()"
```

calls IsValid and submits the form if it returns TRUE, or does not submit the form if it returns FALSE, while:

```
form.onsubmit = "IsValid()"
```

calls IsValid and submits the form regardless of return value.

Applies To

Form

Methods

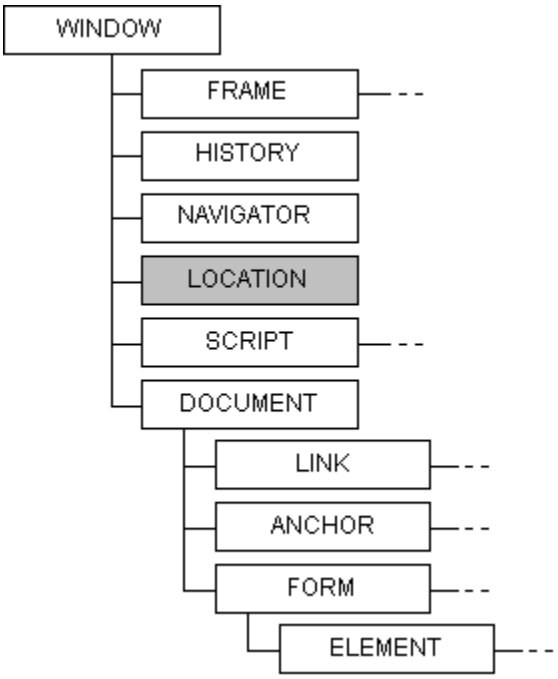
submit

Properties

action, encoding, method, target, elements

location Object

An object that resides below the window in the scripting object model. The location object represents the current URL:



Setting any portion of the location object causes the browser to navigate to the newly constructed URL. The following script navigates to http://www.microsoft.com:

```
<script language="VBScript">
     [some preceding VBScript code]
     location.href="http://www.microsoft.com"
</script>
```

Properties

href, protocol, host, hostname, port, pathname, search, hash

href Property

Gets or sets the compete URL for the location.

Syntax

*location.***href** [**=***string*]

Parts

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the complete URL for the location.

Applies To

Location

Properties

protocol, host, hostname, port, pathname, search, hash

protocol Property

Gets or sets the protocol portion of the URL.

Syntax

location.**protocol** [**=***string*]

Parts

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the protocol portion of the URL.

Remarks

For http://www.microsoft.com, this would return http:.

Applies To

Location

Properties

href, host, hostname, port, pathname, search, hash

host Property

Gets or sets the host and port portions of the URL (hostname:port.).

Syntax

*location.***host** [**=***string*]

Parts

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the host and port portions of the URL.

Remarks

For http://www.microsoft.com:80, this would be www.microsoft.com:80. For file: protocols, this always returns "".

Applies To

Location

Properties

href, protocol, hostname, port, pathname, search, hash

hostname Property

Gets or sets the host portion of the URL, either a name or an IP address.

Syntax

*location.***hostname** [**=***string*]

Parts

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the hostname portion of the URL.

Remarks

For http://www.microsoft.com, this would return www.microsoft.com. For file: protocols, this always returns "".

Applies To

Location

Properties

href, protocol, host, port, pathname, search, hash

port Property

Gets or sets the port of the URL.

Syntax

location.**port** [**=***string*]

Parts

location
An object expression that evaluates to a location object.
string
Optional. The new string value.

Return Values

Returns a string containing the port of the URL.

Remarks

For *http://www.microsoft.com:80*, this returns *80*. For *file:* protocols, this always returns "".

Applies To

Location

Properties

href, protocol, host, hostname, pathname, search, hash

pathname Property

Gets or sets the pathname in the URL.

Syntax

location.**pathname** [**=***string*]

Parts

location
An object expression that evaluates to a location object.
string
Optional. The new string value.

Return Values

Returns a string containing the pathname portion of the URL.

Remarks

For http://www.microsoft.com/intdev, this returns intdev.

Applies To

Location

Properties

href, protocol, host, hostname, port, search, hash

search Property

Gets or sets the search portion of the URL, if specified.

Syntax

location.**search** [**=***string*]

Parts

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the search portion of the URL.

Remarks

For http://www.microsoft.com/intdev?user, this returns ?user. For http://www.microsoft.com/intdev, this returns NULL.

Applies To

Location

Properties

href, protocol, host, hostname, port, pathname, hash

hash

Gets or sets the hash portion of the URL, if specified.

Syntax

*location.***hash** [**=***string*]

Parameters

location
An object expression that evaluates to a location object.

string
Optional. The new string value.

Return Values

Returns a string containing the hash portion of the URL.

Applies To

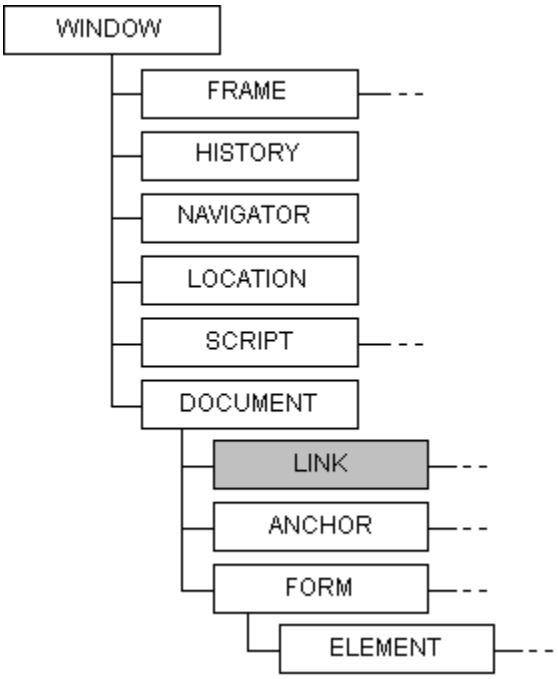
Location

Properties

href, protocol, host, hostname, port, pathname, search

link Object

An object that resides below the document in the scripting object model. This object specifies an array of links for a given document.



The link object is referenced as a read-only property array. A link object is constructed for every link that appears in the HTML document. A link is defined in scripting as the anchor tag <A> containing the HREF attribute, <A HREF="http://www.microsoft.com">. All properties of the link object are read-only and are the same as the location object's properties. It is only accessible through the indexed array. The following lines of script would set linktext to the third link on the page (if it exists):

```
<script language="VBScript">
  [some preceding VBScript code]
  linktext = document.links(2).href
  [some following VBScript code]
</script>
```

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, port, pathname, search, hash, target

href Property

Returns the compete URL for the link.

Syntax

link.**href**[**=string**]

Parts

link
An object expression that evaluates to a link object.

Return Values

Returns a string containing the complete URL for the link.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

protocol, host, hostname, port, pathname, search, hash, target

protocol Property

Returns the protocol portion of the URL.

Syntax

link.**protocol**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the protocol portion of the URL.

Remarks

For http://www.microsoft.com, this would return http:.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, host, hostname, port, pathname, search, hash, target

host Property

Returns the host and port portions of the URL (hostname:port).

Syntax

link.**host**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the host and port portions of the URL.

Remarks

For http://www.microsoft.com, this would return www.microsoft.com:80.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, hostname, port, pathname, search, hash, target

hostname Property

Returns the host portion of the URL, either a name or an IP address.

Syntax

link.**hostname**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the hostname portion of the URL.

Remarks

For http://www.microsoft.com, this would return www.microsoft.com.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, port, pathname, search, hash, target

port Property

Returns the port of the URL.

Syntax

link.**port**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the port of the URL.

Remarks

For http://www.microsoft.com, this returns 80 (the default for HTTP).

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, pathname, search, hash, target

pathname Property

Returns the pathname in the URL.

Syntax

link.**pathname**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the pathname portion of the URL.

Remarks

For http://www.microsoft.com/intdev, this returns /intdev.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, port, search, hash, target

search Property

Returns the search portion of the URL, if specified.

Syntax

link.**search**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the search portion of the URL.

Remarks

For http://www.microsoft.com/intdev?user, this returns user.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, port, pathname, hash, target

hash Property

Returns the hash portion of the URL, if specified.

Syntax

link.**hash**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the hash portion of the URL. If no hash is specified, this property returns NULL.

Remarks

The hash portion of the URL is the section after #, including the #. For http://www.microsoft.com/intdev#user, this returns #user.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, port, pathname, search, target

target Property

Returns the target of the link, if specified.

Syntax

link.**target**

Parts

link

An object expression that evaluates to a link object.

Return Values

Returns a string containing the target of the link.

Remarks

This is the same as the value of the TARGET attribute of the LINK tag.

Applies To

Link

Events

onMouseMove, onMouseOver, onClick

Properties

href, protocol, host, hostname, port, pathname, search, hash

Events

Link events can be used to set status bar text or other custom actions on mouse movement. The following example is an excerpt from an HTML document that uses a text control to display rich information about the links in an image map. The code decides on the link location.

```
<script language="VBScript" for="Link1" event="onMouseMove(shift, button, x,
y)">
    if (InRect(x, y,  5, 30, 120, 85)=true) then
    DescribeLink "A full description of Microsoft's product line"
    [some following VBScript code]
</script>
```

onMouseMove Event Handler

Fires an event any time the pointer moves over a link.

Syntax

link.**onMouseMove** *shift*, *button*, *x*, *y*

Values

link
An object expression that evaluates to a link object.

shift
The status of the shift key.

button
Indicates which button is pressed, if any.

x
The horizontal position of the pointer, in pixels.

y
The vertical position of the pointer, in pixels.

Remarks

Shift and button are currently set to zero. x and y contain the actual positional data. To attach scripts or behavior to this event, use the SCRIPT tag as follows:

```
<script language=script-engine for=link-name event="onMouseMove(shift,
button, x, y)">
```

Applies To

Link

Events

onMouseOver, onClick

Properties

href, protocol, host, hostname, port, pathname, search, hash, target

onMouseOver Event Handler

Fires an event any time the pointer moves over a link.

Syntax

link.**onMouseOver**

Values

link

An object expression that evaluates to a link object. Note: this does not work if the link is inside of a form.

Remarks

To attach scripts or behavior to this event, use the SCRIPT tag as follows:

```
<script language=script-engine for=link-name event="onMouseOver">
```

or attach a script directly in the HTML:

```
<A HREF="http://www.microsoft.com" onMouseOver="alert ('Clicked here')">To Microsoft</A>
```

Applies To

Link

Events

onMouseMove, onClick

Properties

href, protocol, host, hostname, port, pathname, search, hash, target

onClick Event Handler

Fires an event any time you click on a link.

Syntax

link.**onClick**

Values

link

An object expression that evaluates to a link object.

Remarks

To attach scripts or behavior to this event, use the SCRIPT tag as follows:

```
<script language=script-engine for=link-name event="onClick">
```

or attach a script directly in the HTML:

```
<A HREF="http://www.microsoft.com" onClick="alert ('Clicked here')">To
Microsoft</A>
```

Applies To

Link

Events

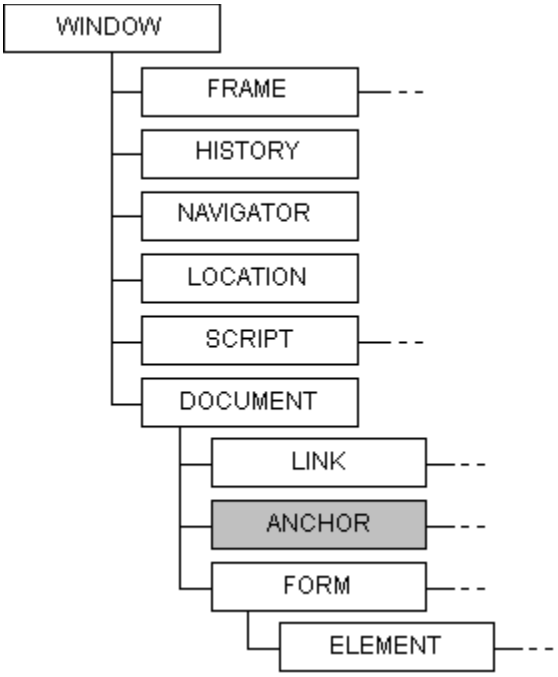
onMouseMove, onMouseOver

Properties

href, protocol, host, hostname, port, pathname, search, hash, target

anchor Object

An object that resides below the document in the scripting object model. This object specifies an array of anchors for a given document. Each entry in this array corresponds to an anchor tag, <A>, that is found in the corresponding document.



The anchor object is referenced as a read-only property array. An anchor object is constructed for every anchor tag, <A>, found in the HTML document. It is only accessible through the indexed array. The following lines of script would set anchortext to the name of the third anchor on the page (if it exists).

```
<script language="VBScript">
     [some preceding VBScript code]
     anchortext = document.anchors(2).name
     [some following VBScript code]
</script>
```

name Property

Gets or sets the name of the anchor.

Syntax

*anchor.***name** [**=***string*]

Parts

anchor
An object expression that evaluates to an anchor object.

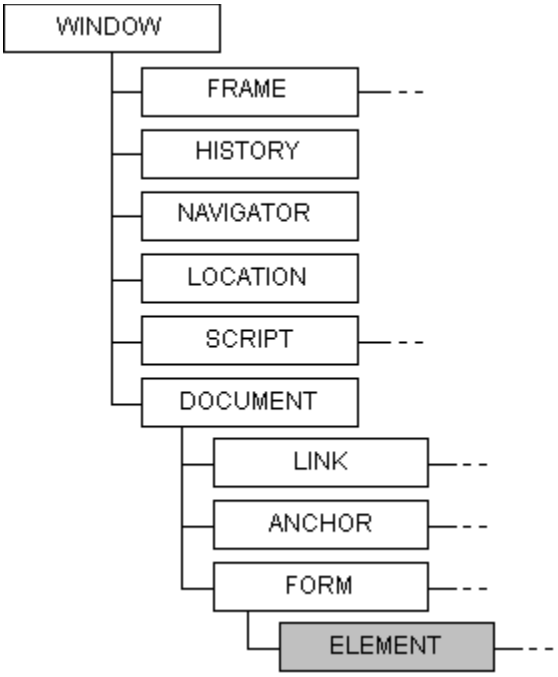
string
A string containing the new anchor name.

Return Values

Returns a string containing the complete name of the anchor.

element Object

An object that resides below the document in the scripting object model. Elements are intrinsic HTML controls or objects. Controls are placed on a document with the <INPUT> tag while objects are placed on a document with the <OBJECT> tag.



Elements are intrinsic HTML controls (placed on a page through the input tag <INPUT>) or objects that are insertable in HTML via the object tag <OBJECT>. These include ActiveX Controls. They can be referenced either by array or name, but this reference must follow the form identifier. Not all properties, methods, and events apply to all elements. Some properties apply to all elements; some only apply to specific elements. See the list below for details by element type, then see the specific method, event, or property documentation for details.

Element	Properties	Methods	Events
button, reset, submit	form, name, value	click	onClick
checkbox	form, name, value, checked, defaultChecked	click	onClick
radio	form, name, value, checked	click, focus	onClick
password	form, name, value, defaultValue	focus, blur, select	
text, textarea	form, name, value, defaultValue	focus, blur, select	onFocus, onBlur, onChange, onSelect
select	name, length, options, selectedIndex	focus, blur	onFocus, onBlur, onChange
hidden	name, value		

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

form Property

Gets the form object containing the element.

Syntax

*element.***form**

Parts

element

An object expression that evaluates to the form containing the element

Return Values

Returns an object expression that evaluates to a form object.

Applies To

All elements except select and hidden.

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

name Property

Gets or sets the name of the element.

Syntax

*element.***name** [**=***string*]

Parts

element
An object expression that evaluates to an intrinsic control.

string
Optional. A string containing the new element name.

Return Values

Returns a string containing the name of the element.

Applies To

All elements

Methods

click, focus, blur, select

Events

onFocus, onBlur, onChange, onSelect

Properties

form, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

value Property

Gets or sets the value of the element.

Syntax

*element.***value** [**=***string*]

Parts

element
An object expression that evaluates to an intrinsic control.

string
Optional. A string containing the new element value.

Return Values

Returns a string containing the value of the element.

Applies To

All elements

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, defaultValue, checked, defaultChecked, length, options, selectedIndex

defaultValue Property

Gets or sets the default value of the element.

Syntax

*element.***defaultValue** [**=***string*]

Parts

element
An object expression that evaluates to an intrinsic control.

string
Optional. A string containing the new default value.

Return Values

Returns a string containing the default value of the element.

Applies To

password, **text**, **textarea**

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, checked, defaultChecked, length, options, selectedIndex

checked Property

Gets or sets the checked state of the checkbox.

Syntax

*element.***checked** [**=***bool*]

Parts

element
An object expression that evaluates to an intrinsic control.

string
Optional. Sets the checked state of the checkbox or the radio button.

Return Values

Returns 1 if the checkbox or radio button is checked; 0 if not.

Applies To

checkbox, **radio button**

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, defaultChecked, length, options, selectedIndex

defaultChecked Property

Gets or sets the default checked property of the checkbox.

Syntax

*element.***defaultChecked** [**=***bool*]

Parts

element
An object expression that evaluates to an intrinsic control.

string
Optional. Sets the default state of the checkbox.

Return Values

Returns TRUE if the checkbox is checked by default; FALSE if not.

Applies To

checkbox

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, length, options, selectedIndex

length Property

Gets the number of options in a select element.

Syntax

*element.***length**

Parts

element

An object expression that evaluates to a select element.

Return Values

Returns an integer specifying the number of options in a select element.

Applies To

select

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, options, selectedIndex

options Property

Gets the <options> tags for a select element.

Syntax

*element.***options**

Parts

element
An object expression that evaluates to a select element.

Return Values

Returns an object with the <options> for a select element.

Remarks

The options array has the following properties:

defaultSelected	Identifies the currently selected attribute.
index	Specifies the index of an option.
length	Specifies the number of options in the selected object.
name	Specifies the name attribute of the selected object.
selected	Used to programmatically select an option.
selectedIndex	Specifies the index of the selected option.
text	Specifies the text to be displayed (this text follows the <option> tag).
value	Specifies the value attribute.

Applies To

select

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, selectedIndex

selectedIndex Property

Gets the index for the selected option (or the first option selected when there are multiple selected options).

Syntax

*element.***selectedIndex**

Parts

element
An object expression that evaluates to a select element.

Return Values

Returns an integer specifying the index for the selected option in a select element.

Applies To

select

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options

click Method

Clicks the element.

Syntax

*element.***click**

Parts

element

An object expression that evaluates to an intrinsic control.

Applies To

button, **reset**, **submit**, **checkbox**, **radio**

Methods

focus, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

focus Method

Sets the focus to the element.

Syntax

*element.***focus**

Parts

element
An object expression that evaluates to an intrinsic control.

Applies To

password, **text**, **textarea**, **select**

Methods

click, blur, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

blur Method

Clears the focus from the element.

Syntax

*element.***blur**

Parts

element
An object expression that evaluates to an intrinsic control.

Applies To

password, **text**, **textarea**, **select**

Methods

click, focus, select

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

select Method

Selects the contents of the element.

Syntax

*element.***select**

Parts

element

An object expression that evaluates to an intrinsic control.

Applies To

password, **text**, **textarea**

Methods

click, focus, blur

Events

onClick, onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

Event Handlers

There are two ways to script events from objects:

1 Using the onEvent="subroutine" syntax. This method can be used for any HTML intrinsic elements, such as forms, buttons, or links. This method does not work for items inserted using the OBJECT tag. The following example uses this syntax in Button1 to handle onClick:

```
<form name="Form1">
     <input type="button" name="Button1" value="Press me" onClick="pressed">
</form>

<script language="VBScript">
     sub pressed
     alert "I've been pressed"
     document.Form1.Button1.value="OUCH"
     end sub
</script>
```

2 Using the FOR="object" EVENT="eventname" syntax. This method can be used for any named element and any element inserted using the OBJECT tag. The following example is the same as the first but with a different syntax:

```
<form name="Form1">
     input type="button" name="Button1" value="Press">
     <script for="Button1" event="onClick" language="VBScript">
     alert "Button pressed"
     document.Form1.Button1.value="Pressed"
     </script>
</form>
```

onClick Event Handler

Fired when the element is clicked.

Syntax

*element.***onClick**

Values

element

An object expression that evaluates to an intrinsic control.

Applies To

button, **reset**, **submit**, **checkbox**, **radio**

Methods

click, focus, blur, select

Events

onFocus, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

onFocus Event Handler

Fired when the element gets the focus.

Syntax

*element.***onFocus**

Values

element

An object expression that evaluates to an intrinsic control.

Applies To

select, **text**, **textarea**

Methods

click, focus, blur, select

Events

onClick, onBlur, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

onBlur Event Handler

Fired when the element loses the focus.

Syntax

*element.***onBlur**

Values

element

An object expression that evaluates to an intrinsic control.

Applies To

select, **text**, **textarea**

Methods

click, focus, blur, select

Events

onClick, onFocus, onChange, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

onChange Event Handler

Fired when the element has changed.

Syntax

*element.***onChange**

Values

element

An object expression that evaluates to an intrinsic control.

Applies To

select, **text**, **textarea**

Methods

click, focus, blur, select

Events

onClick, onFocus, onBlur, onSelect

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

onSelect Event Handler

Fired when the contents of the element are selected.

Syntax

*element.***onSelect**

Values

element

An object expression that evaluates to an intrinsic control.

Applies To

text, **textarea**

Methods

click, focus, blur, select

Events

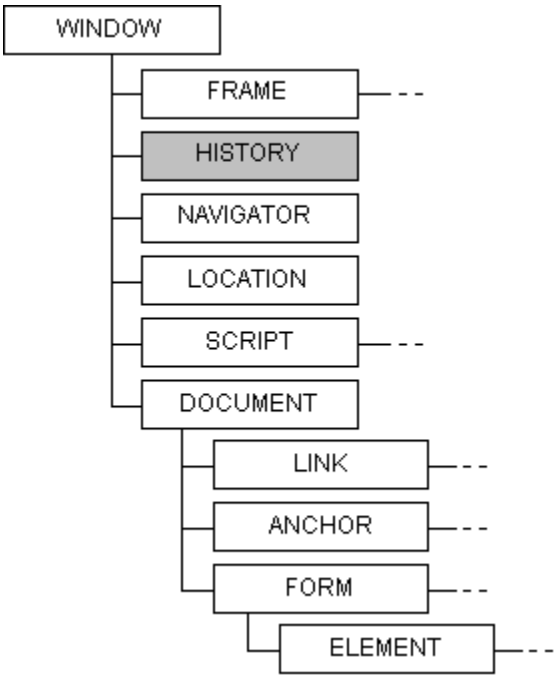
onClick, onFocus, onBlur, onChange

Properties

form, name, value, defaultValue, checked, defaultChecked, length, options, selectedIndex

history Object

An object that resides below the window in the scripting object model. This object accesses the history list from the browser.



The history object exposes methods for navigating through the current history.

Methods

back, forward, go

Properties

length

length Property

Returns the length of the history list.

Syntax

*history.***length**

Parts

history
An object expression that evaluates to a history object.

Return Values

Returns the number of entries in the history.

Always returns zero in current implementation.

Applies To

History

Methods

back, forward, go

back Method

Jumps back in the history *n* steps. This behaves exactly as if the user has clicked on the back button *n* times.

Syntax

*history.***back** *n*

Parts

history
An object expression that evaluates to a history object.

n
The number of pages to jump back in the history. *n* is always an integer >= 0.

Applies To

History

Methods

forward, go

Properties

length

forward Method

Jumps forward in the history *n* steps. This behaves exactly as if the user has clicked on the forward button *n* times.

Syntax

*history.***forward** *n*

Parts

history
An object expression that evaluates to a history object.

n
The number of pages to jump forward in the history. *n* is always an integer >= 0.

Applies To

History

Methods

back, go

Properties

length

go Method

Goes to the *n* th item in the history, where *history.***go** 1 jumps to the first item in the history.

Syntax

*history.***go** *n*

Parts

history
An object expression that evaluates to a history object.

n
The index of the history entry.

Applies To

History

Methods

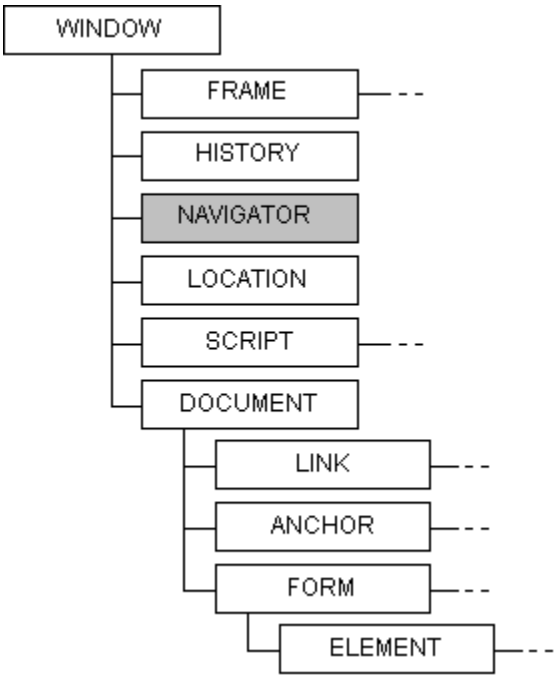
back, forward

Properties

length

navigator Object

An object that resides below the window in the scripting object model. The navigator object provides information about the browser in use.



Properties

appCodeName, appName, appVersion, userAgent

appCodeName Property

Returns the code name of the application.

Syntax

*navigator.***appCodeName**

Parts

navigator
An object expression that evaluates to a navigator object.

Return Values

Returns a string containing the current application code name.

Applies To

Navigator

Properties

appName, appVersion, userAgent

appName Property

Returns the name of the application. Internet Explorer 3.0 currently returns "Microsoft Internet Explorer."

Syntax

*navigator.***appName**

Parts

navigator
An object expression that evaluates to a navigator object.

Return Values

Returns a string containing the current application name.

Applies To

Navigator

Properties

appCodeName, appVersion, userAgent

appVersion Property

Returns the version of the application.

Syntax

*navigator.***appVersion**

Parts

navigator
An object expression that evaluates to a navigator object.

Return Values

Returns a string containing the current application version.

Applies To

Navigator

Properties

appCodeName, appName, userAgent

userAgent Property

Returns the user agent of the application. Internet Explorer 3.0 currently returns "Mozilla/2.0 (compatible; MSIE 3.0A; Windows 95)".

Syntax

*navigator.***userAgent**

Parts

navigator
An object expression that evaluates to a navigator object.

Return Values

Returns a string containing the current application user agent.

Applies To

Navigator

Properties

appCodeName, appName, appVersion