

Internet Explorer 4.0 Technical Articles

An Extended Object Map of Internet Explorer 4.01

Kenneth Lassesen
Microsoft Corporation

February 1998

Microsoft® Internet Explorer is the world's Automation browser of choice. Developers can quickly build customized browsers using its browser control inside an application developed with Visual Basic® or Visual C++® development systems. America Online (AOL), MSN™, The Microsoft Network, and other firms provide customized browsers, built around this control to customers.

Internet Explorer may be automated as an in-process control in an application called the Microsoft Web Browser Control (WebBrowser) or as an out-of-process, stand-alone application in its own window called Microsoft Internet Explorer (IWebBrowser2). Both objects originate in the SHDocVw.dll with an overloading of properties that can become confusing. Both IWebBrowser2 and WebBrowser objects have a **FullScreen** property that may be set or read. In both, setting the value will trigger an onFullScreen event. WebBrowser does not react to this property's value. IWebBrowser2 will become full screen.

WebBrowser is a control that has some extended object properties and methods shared by all controls, such as: **Move**, **Zorder**, and **ShowWhatsThis**. Some properties in IWebBrowser2 are replaced with the extended object properties—for example, **Left**, **Top**, **Width**, and **Height**. With IWebBrowser, the values of **Left**, **Top**, **Width**, and **Height** are *Long* values typical for a stand-alone application while with the WebBrowser they are *Single* values typical for a control.

The change of event models causes a further complication. The Internet Explorer 3.0 events (DwebBrowserEvents) are available in Internet Explorer 4.0 for existing compiled applications but are not available in the Visual Basic integrated development environment (IDE). Care must be taken modifying applications using Internet Explorer events, as Table 1 summarizes:

Compiled using:	Some Events will not fire in the following browsers:	All Events work with:
Internet Explorer 3.0		Internet Explorer 3.0
		Internet Explorer 4.0
Internet Explorer 4.0	Internet Explorer 3.0	Internet Explorer 4.0

Table 1. Browser event compatibility

If you are developing an application that will be used with both Internet Explorer 3.0 and Internet Explorer 4.0, remember to use Internet Explorer 3.0 for development because Internet Explorer 3.0 is not forward compatible with Internet Explorer 4.0 events. The map shows how the unique Internet Explorer 3.0 events correspond to the new events. The arguments are not the same, as is shown in the following example:

```
Internet Explorer 3.0 Event
```

```
Private Sub WebBrowser1_NavigateComplete(ByVal URL As String)
```

```
Internet Explorer 4.0 Event
```

```
Private Sub WebBrowser1_NavigateComplete2(ByVal pDisp As Object, URL As Variant)
```

With most Automation servers, there are distinct **print**, **copy**, and **select all** methods. The methods do not exist as distinct methods with Internet Explorer 4.0. If you wish to print, copy, or select all through Automation, you call the **ExecWB** method with appropriate arguments. The **QueryStatusWB** is used to see if these methods are available in the current context (for example, you may not be able to print while the page is being downloaded).

Internet Explorer has a **Document** property that connects to whatever is being displayed in

Internet Explorer. This may be a Microsoft Word document, a Microsoft Excel spreadsheet, or, typically, an HTML page. An HTML page is represented by the Microsoft HTML Object Library, which is familiar to Web-design engineers and others who use scripting with HTML. The Microsoft HTML Object Library is a very rich object and only the top layer is shown on the map. A few examples of its possible use are shown here.

The following lines write HTML directly from Visual Basic into the WebBrowser:

```
'WebBrowser must first navigate to a HTML file.
WebBrowser1.Document.Open
WebBrowser1.Document.Write "<H1>Hello world</H1>"
WebBrowser1.Document.Close
```

You are able to dynamically create controls on the fly for use from your compiled application. The following sample creates and uses the calendar control:

```
'WebBrowser must first navigate to a HTML file.
WebBrowser1.Document.Open
WebBrowser1.Document.Write "<object id=calendar classid=clsid:8E27C92B-1264-101C
WebBrowser1.Document.Close
msgbox "The month is:" + WebBrowser1.Document.calendar.month
```

The final example shows you how to create an evaluate function, dynamically create JavaScript objects, or write code that writes code that writes code that . . . :

```
'WebBrowser must first navigate to a HTML file.
WebBrowser1.Document.Open
WebBrowser1.Document.Write "<Script> function evaluate(x){return eval(x)}</SCRIPT>"
WebBrowser1.Document.Close
msgbox WebBrowser1.Document.script.evaluate("5+Math.sin(9)")
' 5.41211848524176 appears like magic
```

In short, Internet Explorer is an awesome control that provides features that extend Visual Basic or Visual C++.

Figure 1 below represents graphically an extended object map of Internet Explorer 4.01.

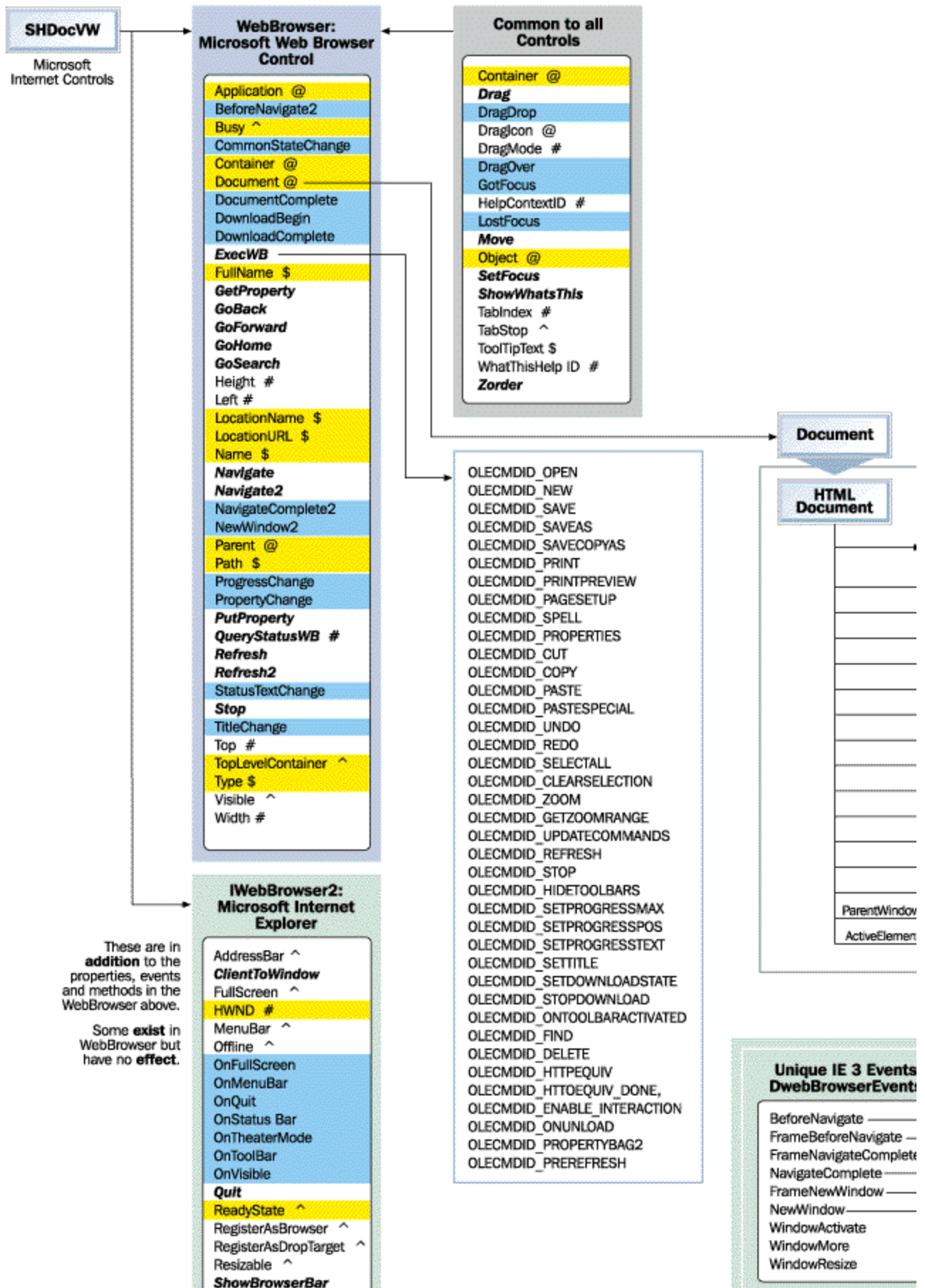


Figure 1. Extended object map of Internet Explorer 4.01

[Send feedback to Microsoft](#)

[© 2003 Microsoft Corporation. All rights reserved.](#)