# Taking Advantage of Internet Explorer

*How to make use of the Internet Explorer object in VBScript*

by Bob Kelly
January 2007

When you think of making use of Internet Explore in your scripts, you probably think of using a hypertext application (HTA). HTA scripts are web documents with script code built in. However, you can actually take advantage of IE in your VBScripts without using HTA. Although you don't have the same interactive options, it can be a nice way to display information (progress for example) to users outside of the standard console window. Here's how…

Start by creating a reference to the Internet Explorer Application object:

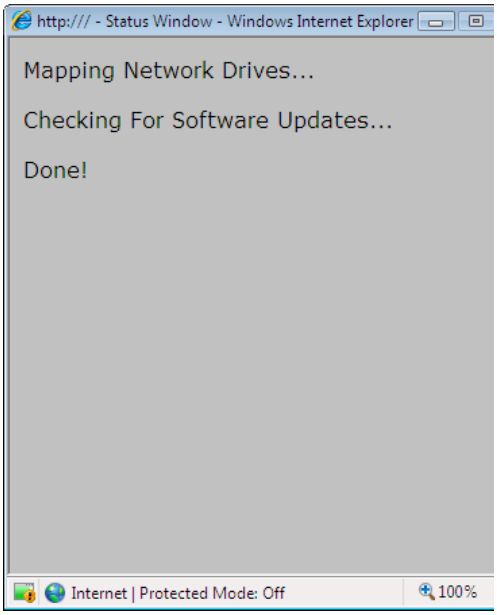
```
Set oIE =
CreateObject("InternetExplorer.Application")
```

In an HTA script, you would typically establish the details of the window in the HTA:APPLICATION tag. Similarly, we can set many of these same properties using the IE object…

```
oIE.AddressBar = False
oIE.Menubar = False
oIE.Toolbar = False
oIE.Resizable = False
oIE.Height = 450
oIE.Width = 400
oIE.Visible = True
```

We need to tell IE to open a page, but because we are not actually creating a file for it to load, set the navigation to load a blank page…

```
oIE.Navigate("about:blank")
```




Now not everyone has as fast as a machine as I'm sure you are sitting at right now, so it is a good move to have your script wait until IE is ready to respond to your script. We do this by sleeping until IE no longer reports itself as "busy"…

```
While oIE.Busy
      WScript.Sleep 100
Wend
```

Now that IE is ready for us, we need to establish another object—this time for the document with which we will be working. Again, this is not an actual file, so once opened, any information to be displayed may be written to the document object in HTML format and it will be displayed in real-time. As an example, This script reports that is mapping drives and then calls a subroutine where you could map some drives (but for the sake of keeping on focus, I've simply provided a comment and have the script sleep so as to simulate actual work being done. The same is done for a

software update check and you could obviously replace these with whatever operations you wish to show status on. However, this should be fairly easy to read and make your own should you find the need…

```
Set oDoc = oIE.Document
oDoc.Open

oDoc.Write("<TITLE>Status Window</TITLE>")
oDoc.Write("<BODY BGCOLOR=#C0C0C0>")
oDoc.Write("<P><FONT FACE=""Verdana"">Mapping Network " &_
           "Drives...</FONT></P>")
MapDrives()
oDoc.Write("<P><FONT FACE=""Verdana"">Checking For " &_
           "Software Updates...</FONT></P>")
SoftCheck()
oDoc.Write("<P><FONT FACE=""Verdana"">Done!</FONT></P>")

WScript.Sleep 1800
oIE.Quit
WScript.Quit

Sub MapDrives
      'Map drives here
      WScript.Sleep 900
End Sub

Sub SoftCheck
      'Check for software updates here
      WScript.Sleep 900
End Sub
```

So there we have it: another alternative to GUI scripting and again without using a HTA. Next time, I'll put some focus on this mysterious HTA I keep talking around. There are some cool things that can be done with GUI scripts and HTA has some nice benefits too!

*Bob Kelly is president and co-founder of AdminScriptEditor.com, home to an integrated suite of scripting tools and a shared library of scripts and language help. He has authored books on scripting and desktop administration as well as several white papers. Bob also owns and operates AppDeploy.com, where he writes and produces videos on topics related to software deployment. You can contact Bob about "GUI Scripts with KiXforms" at bkelly@adminscripteditor.com.*