Notes about VBScript

Gene Michael Stover

updated 5 October 2003

Contents

1 Introduction

These are random, informal, & not necessarily informative notes about Microsoft's Visual Basic Script (VBScript).

2 Syntax & Semantic Gotchas

2.1 Strictly Defined Variables

To make VBScript complain if you forget to declare a variable, use "`Option Explicit`" near the beginning of your program.

2.2 Assignments

To assign an object reference to a variable, you must use `Set`. Like this:

```
REM Declare the variable because it's good form.
Dim fso

REM Create a File System Object & bind it to
REM 'fso'.  We must use Set, not a plain
REM assignment.
Set fso = CreateObject ("Scripting.FileSystemObject")
```

If you use a plain assignment to bind an object reference to a variable, you'll actually bind Null to the variable. You won't get a syntax error or a run-time error, so it's a fairly insidious problem.

To assign a datum other than an object reference to a variable, use a plain assignment, like this:

```
Dim str

str = "Hello " & "there."
```

If you attempt to assign a non-object datum to a variable with a `Set`, you'll get a run-time error.

2.3 Standard I/O

You can read & write the standard I/O streams from VBScript if `cscript` is your VBScript interpreter. (It doesn't work for `wscript`.) Here's how:

```
REM Declare our variables, to be in good form.
Dim fso, stdin, stdout, stderr

REM Need to create a File System Object.  It
REM seems that a lot of programs need a File
REM System Object.
Set fso = CreateObject ("Scripting.FileSystemObject")

REM Finally, just get the streams & bind them
REM to those variables we created.
REM Notice that the argument to GetStandardStream
REM corresponds to the Unix/POSIX file
REM descriptors.
Set stdin = fso.GetStandardStream (0)
Set stdout = fso.GetStandardStream (1)
Set stderr = fso.GetStandardStream (2)

REM Now we can write to stdout or stderr, or
REM we can read from stdin.  Here's an example
REM of output.
stdout.WriteLine "Hello, VBScript."
stderr.WriteLine "I could write an error here."
```

This doesn't work when `wscript` is your interpreter. With `wscript`, `FileSystemObject.GetStandardStrea` returns Null, so you'll get an error when you try to use the stream, not when you create it. If you run a `*.vbs` program from the command line by typing its name, you'll probably get `wscript` as your interpreter. If you want to use standard I/O streams, you must run your `*.vbs` program explicitly with `cscript`.

3 Fun with Visual Basic Script & Internet Explorder

3.1 vbsie000.html

vbsie000.html is a Visual Basic Script program that prints some hard-coded paragraphs to Document in `window_onload`. The idea is to see where the text printed to the Document fits into the Web page. Does it go in the HEAD? At the beginning of the BODY? (`vbsie000.html` suggests that it replaces the entire BODY. Also, it messes up the browser's history list.)

3.2 vbsie001.html

vbsie001.html is like vbsie000.html except that it creates a Window with `Document.open` & writes the hard-coded HTML paragraphs to that. (Running vbsie001.html suggests that text written to the new window doesn't show on the browser. What's more, the text that was hard-coded into the HTML page does not show.)

vbsie002.html Uses nested loops in VBS to write ten paragraphs of lots of words (which are just numbers). Each word is transmitted on a line by itself.

vbsie003.html. During the window's `OnLoad`, displays a message box. Visiting vbsie003.html demonstrates that message boxes do not mess up the browser's history list. Also, vbsie003.html does not print anything to the Document object, so the regular HTML text in that page displays correctly.

vbsie004.html shows a message box with the `Window.clientInformation` property. It turns out that the client information property isn't very interesting. When I've run it, it just holds &

$$objectNavigator$$

&.

vbsie005.html Creates a new window with `Window.open`, except the the VBS program is part of the Window, so I just call "`open ()`". vbsie005.html does indeed create a new window, though it doesn't put any text into it.

vbsie006.html tries writing "Hello, window" to the new window's Document. Still doesn't print anything in the new window. In my VBS book (page 169), it says you must call Document.write when the window is created. So maybe what you must do is open the new window to an URL that contains a web page. The URL's page might even contain a VBS program called `window_onload`.

vbsie007.html

End.