



JavaScript Mini-FAQ

By Danny Goodman

All materials Copyright © 1997–2002 Developer Shed, Inc. except where otherwise noted.

This Mini-FAQ is posted periodically to the comp.lang.javascript newsgroup. It covers the language through JavaScript 1.2, the version deployed in Netscape Communicator 4.0x, plus some compatibility items with Microsoft Internet Explorer 3.0x. The focus here is on client-side JavaScript.

Where is the online documentation for JavaScript?

Current JavaScript docs (for Netscape) are available at:

- <http://home.netscape.com/eng/mozilla/3.0/handbook/javascript/index.html>

A zipped set of Netscape's HTML documents is available at:

- <http://developer.netscape.com/library/documentation/jshtm.zip>

New JavaScript features in Netscape Communicator can be found at:

- http://developer.netscape.com/library/documentation/communicator/jsguide/js1_2.htm

Documentation for Microsoft's implementation of its core language (called JScript) is at:

- <http://www.microsoft.com/JScript/us/techinfo/jsdocs.htm>

Also be sure to download Microsoft's document object model description. You can find a link from:

- <http://www.microsoft.com/JScript/>

Documentation for JScript in Internet Explorer 4 is part of Microsoft's Internet Client SDK documentation:

- <http://www.microsoft.com/msdn/sdk/inetsdk/asetup/>

Where is the official bug list for JavaScript?

Netscape has collected and published a list of bugs for Navigator 3.0x and Communicator. While not necessarily 100% complete, it is quite extensive:

- <http://developer.netscape.com/>

Can JavaScript do any of the following?

- **read or write random text files on the local disk or on the server?**
- **invoke automatic printing of the current document?**
- control browser e-mail, news reader, or bookmark windows and menus?
- **access or modify browser preferences settings?**
- capture a visitor's e-mail address or IP address?
- quietly send me an e-mail when a visitor loads my page?
- **launch client processes (e.g., Unix sendmail, Win apps, Mac scripts)?**
- **capture individual keystrokes?**
- change a document's background .gif after the page has loaded?
- **change the current browser window size, location, or options?**
- **get rid of that dumb "JavaScript Alert:" line in alert dialogs?**

No, however many of these items are possible in **Communicator 4.0**. Those items perceived to be security risks (e.g., access browser settings) require "signed JavaScript". MSIE JScript version 2 (see below) can read/write local files via ActiveX--but only from server-side scripting.



JavaScript Mini-FAQ

Why won't my script work under MS Internet Explorer 3 for the Mac? JScript is available on the Macintosh starting with 3.0.1 (which is different from the Windows 3.01). I am still evaluating the Mac implementation, whose object model and other support for JavaScript does not necessarily jive with the Windows version (e.g., the Mac version supports the Image object for mouse rollovers). MSIE 3.0.1 runs on Mac 68K and PPC.

Why won't my Navigator 3.0x script run under MSIE 3 for Windows 95?

Most language features and objects that are new in Navigator 3.0 are not supported in MSIE 3.0, although several Navigator 3.0 items have been added to JScript version 2 (see below). Here's the quick list of items not available in MSIE 3.0:

UNSUPPORTED OBJECTS

- Image -- this means no onMouseOver swappable images in MSIE 3
- Area -- no onMouseOvers
- Applet
- FileUpload
- Array -- hard-wired (JS1.0) arrays OK; implemented in JScript v.2.
- MimeType
- Plugin

UNSUPPORTED PROPERTIES / METHODS / EVENT HANDLERS OF SUPPORTED OBJECTS

- Window
onerror closed blur() focus() scroll() onBlur= onFocus=
- Location
reload() replace()
- Document
applets[] domain embeds[] images[] URL
- Link
onMouseOut=
- Form
reset() onReset=
- (All Form Elements)
type
- Navigator
mimeTypes[] plugins[] javaEnabled()
- String
prototype split()

One more item: the <SCRIPT SRC="xxx.js"> facility for loading external JavaScript library files runs on the copy of MSIE 3.02 for Windows that I use (with JScript.dll versions 1 and 2). However there are also reports that this is not working for some users. Try specifying a complete URL for the SRC attribute.

How is compatibility with Microsoft Internet Explorer 4?



JavaScript Mini-FAQ

IE4 adheres closely to a standard called ECMAScript, which is essentially the core JavaScript 1.1 language. This does not cover the document object model (another standard being studied). Navigator 3 document objects not supported in IE4 are:

`FileUpload` `navigator.mimeTypes[]` `navigator.plugins[]`

The JScript.dll shipping with IE4 is version 3.

Why doesn't the `document.cookie` work with MSIE?

It does, but not when you access the HTML file from your local hard disk, as you are probably doing during testing. Be aware, however, that MSIE limits you to one cookie name=value pair per domain, whereas Netscape allows up to 20 pairs per domain.

What's new in Microsoft JScript version 2?

More than can fit here. Some items are compatible with Navigator 3.0+ (such as the Array object). Others are unique to MSIE, such as the Dictionary and TextStream objects (accessible via ActiveX). Additions are to the core language, not the document object model. New functions let you determine the JScript version installed in IE, but JScript version 2 must be installed to get this data. If you use version 2 language items, see:

- <http://www.microsoft.com/JScript/>

for info about including a link button on your page to encourage visitors to upgrade their IE 3.0x to JScript version 2.

How do I know if I have JScript version 2 installed on my PC?

Installation of MSIE 3.02 does not guarantee JScript version 2. Search your disk for 'jscript.dll'. Get the file's properties, and click on the Version tab. The File version should begin with '2'. If not, download the latest version from Microsoft (installer is 442KB).

How can I e-mail forms?

The most reliable way is to use straight HTML via a Submit style button. Set the ACTION of the <FORM> to a `mailto:` URL and the ENCTYPE attribute to "text/plain". For security reasons, the `form.submit()` method does not submit a form whose ACTION is a `mailto:` URL. Microsoft Internet Explorer 3.0x does not e-mail forms of any kind.

How do I script a visit counter?

At best, a client-side script can show the visitor how many times he or she has been to the site (storing the count in a local cookie). A count of total hits to the server requires a server-side CGI program. I have an article on cookies in Netscape's [View Source](#) developer newsletter archive (in the "JavaScript Apostle" section).

Why is my script not working inside a table?

There is a long-standing bug with JavaScript and tables. Do not place <SCRIPT> tags inside <TD> tags. Instead, start the <SCRIPT> tag before the <TD> tag, and `document.write()` the <TD> tag through the </TD> tag. I go one step further, and `document.write()` the entire table, interlacing script statements where needed.



After `window.open()`, how do I access objects and scripts in the other window?

First, be sure to assign an 'opener' property to the new window if you are using a version of JS that doesn't do it automatically (Nav 3.0x and MSIE 3.0x do it automatically). The following script should be a part of `_every_` new window creation:

```
var newWind = window.open("xxx", "xxx", "xxx")
// u fill in blanks
if (newWind.opener == null) { // for Nav 2.0x
    newWind.opener = self // this creates and sets a new property
}<
```

To access items in the new window from the original window, the 'newWind' variable must not be damaged (by unloading), because it contains the only reference to the other window you can use (the name you assign as the second parameter of `open()` is not valid for scripted window references; only for TARGET attributes). To access a form element property in the new window, use:

```
newWind.document.formName.elementName.property
```

From the new window, the 'opener' property is a reference to the original window (or frame, if the `window.open()` call was made from a frame). To reference a form element in the original window:

```
opener.document.formName.elementName.property
```

Finally, if the new window was opened from one frame in the main browser window, and a script in the new window needs access to another frame in the main browser window, use:

```
opener.parent.otherFrameName.document.formName. . .
```

How do I use JavaScript to password-protect my Web site?

There are any number of schemes (I've used some myself). Most of them fail to deflect the knowledgeable JavaScript programmer, because no matter how you encode the correct password (e.g., bit shifting), both the encoding algorithms and the result have to be in the script -- whose source code is easily accessible. If you're only interested in keeping out casual visitors, this method may suffice.

A more secure way is to set the password to be the name or pathname of the HTML file on your site that is the 'true' starting page. Set the location to the value entered into the field (unfortunately, you cannot extract the value property of a password object in Navigator 2.0x). Entry of a bogus password yields an 'invalid URL' error.

If the protected pages need additional security (e.g., an infidel has managed to get the complete URL), you might also consider setting a temporary cookie on the password page; then test for the existence of that cookie upon entry to every protected page, and throw the infidel back to the password page.

What does the IE4 "Access Denied" error mean when accessing a new window?

The "Access Denied" error in any browser usually means that a script in one window or frame is trying to

JavaScript Mini-FAQ

access another window or frame whose document's domain is different from the document containing the script. What can seem odd about this is that you get this error in IE4 frequently when a script in one window generates a new window (with `window.open()`), and content for that other window is dynamically created from the same script doing the opening. The `focus()` method also triggers the error.

In my experience, this occurs only when the scripts are being run from the local hard disk. You get a clue about the situation in the titlebar of the new window: It forces an `about:blank` URL to the new window, which is a `protocol:domain` that differs from wherever your main window's script comes from. If, however, you put the same main window document on a server, and access it via `http:`, the problem goes away.

There is a workaround for the local-only problem: In the first parameter of the `window.open()` method call, load a real document (even if it is a content-free HTML document) into the sub-window before using `document.write()` to generate content for the subwindow. The loading action 'legitimizes' the window as coming from the same domain as your main window's document.

(This solution does not affect scripts that load a page from a secure server into a separate window or frame. An `http:` protocol in one window and `https:` in the other—even if from the same `server.domain`—yield a security mismatch and "Access Denied." Setting the `document.domain` properties of both pages may solve the problem (but I am unable to test it for sure).)

