**The window.open() Method Revisited**

The window.open() method RETURNS a reference to the new browser window ITSELF. If you store this REFERENCE to the new browser window in a variable, you can then manipulate the actual new browser window in many ways, via appropriate methods. Let's look at ONE of these methods, focus().

The focus() method of the Window object allows us to bring a browser window into FOCUS; that is to say, it allows us to SELECT the browser window without physically touching it, bringing that browser window in front of everything else.

When you call the window.open() method, you must place the returned window reference into a variable, either global or local. Here's an example of what I mean.

Example:

function doStuff() {

var myWindow = window.open("http://www.yahoo.com/", "marian",

"width=300,height=200");

}

Once you have the reference to the new window in a variable, you can access methods of that particular window via the variable name. For instance, after the window has been opened, we will FOCUS the window by calling the focus() method of the window instance of the Window object, using dot-syntax to do so.

Example:

function doStuff() {

var myWindow = window.open("http://www.yahoo.com/", "marian",

"width=300,height=200");

myWindow.focus();

}

Using dot-syntax, I have called the focus() method of the variable, myWindow, which contains the reference to "marian", the new browser window; by doing so, I can bring "marian" forward. This can be important if the window "marian" has already been brought into existence, and has been deselected without closing it. If a window has been deselected without being closed, it gets blurred and sent behind the main browser window; when a method, like doStuff() above, opens "marian" again, the new URL and appearance information is applied to the existing "marian" window, but most web browsers will NOT focus "marian", bringing her forward, unless the focus() method has been applied to her.

Of course, the most EXCITING thing about having a direct JavaScript reference to a browser window is that you can then send HTML, CSS, JavaScript, or other information into that new browser window, creating many dynamic possibilities (all of which, sadly, are beyond the scope of this class).

Due to security restrictions in JavaScript, the CURRENT browser window can NOT be manipulated in the same way that a NEW browser window (created by the window.open() method) can. This restriction is in place to prevent JavaScript programmers from messing with a user's browser window without their permission, a very reasonable precaution. As I have mentioned, however, a NEW browser window, one which YOU have created, is not subject to these restrictions, and can be moved around, changed in dimension, brought forward and back, and many other things, all using built-in JavaScript methods accessed via a window reference in a variable, like the one in the example above.

Although I have not created an exercise for this section, I strongly urge you to experiment with some of these methods, to get comfortable with accessing your window via a variable containing the window's object reference. Remember that you may place that reference within either a local OR a global variable.

I hope that I have whetted your appetite a little bit. Next, we're going to look at custom status bar messages; these also require the manipulation and passing of returned values, although in a considerably different manner.