JavaScript for Beginners

March 15, 2012

WHAT IS JAVASCRIPT?

JavaScript allows you to create 'dynamic, interactive' web pages: web pages that do things in response to what your readers do, without having a separate page for each possible response. In this tutorial, I'm going to cover some of the very basics of using JavaScript in your web pages. I'm going to assume that you know a little bit about HTML, but nothing about programming or JavaScript. If you aren't familiar with HTML, you'll want to go over the Web Writing Guide at http://www.hoboes.com/NetLife/Web_Writing/ first.

First, JavaScript has nothing to do with Java. Java is a separate programming language. JavaScript is a language for scripting web browsers and web servers, although it is seeing other uses as it becomes popular. Java was created by Sun and is designed for complete software applications, while JavaScript was created separately by Netscape Corporation and was later standardized as the ECMAScript standard. It is designed for modifying (and "scripting") web pages.

WHAT IS THAT COVER?

The cover image is a tracing, done in Inkscape, of Mount Semeru and Mount Bromo on the Isle of Java. Java may mean "beyond", but it might also mean "home". Java was created through volcanic activity and still contains many active volcanoes. Despite these dangers, it is the most populous island in Indonesia. Volcanic activity ensures a rich soil throughout the island.

Whether any of this relates to JavaScript is left to your imagination.

INTERCEPTING CLICKS

JavaScript turns every web page into a collection of objects that you can modify and interact with. JavaScript turns everything that the visitor does into events that you can modify and intercept.

Go to http://www.hoboes.com/NetLife/Web_Scripting/Web_Scripting_Resources/ to download the resource zip archive and uncompress it. Open the file "1941.html" in your favorite text editor. There are several links on that page; some of the links are local (they go to hoboes.com) and some of them are not local (they go to Wikipedia and Amazon). We're going to make the non-local ones open in a new window (or tab, depending on the visitor's preferences).

Look in the HTML's "head" area for a comment that says "put your scripts.js here". Underneath that line, add this:

```
<script type="text/javascript" src="scripts.js"></script>
```

This line tells the browser to look for some scripts in the file "scripts.js". We're going to put our JavaScript code there. The syntax for the <script> tag is very similar to the syntax for the <img> tag.

The <script> tag can go anywhere in your document, but you'll usually put it in the <head> area when you're using it to include a file containing JavaScript. We'll use it for other purposes later in this tutorial.

BLOCK ALL LINKS

Our first version will do nothing—literally. It will block all links on this page. Put this into your scripts.js file:

```
function manageLinks(event) {
  return false;
}

document.onclick=manageLinks;
```

Save scripts.js into the same folder as 1941.html. Reload 1941.html in your browser. Try clicking on some of the links: none of them should work.

If the links do work, you're going to need to track down the error. In Firefox or in Safari, you have an error console that will display any errors in your page, as well as the line those errors occurred on. If you're not already using one of those browsers, you'll need to download the latest version and use it for testing your scripts.

Functions

What do these four lines do? The first three lines create a “function”. A function is a collection of scripting lines that you can call by name elsewhere in your script file and anywhere on your web page. We’ll be creating a lot of functions over the course of this tutorial. This function has the name “manageLinks”. The lines of a function are contained between the opening curly bracket and the closing curly bracket.

Functions often accept “parameters” or “arguments”. The manageLinks function accepts one parameter, which it names “event”.

Functions often return data to whoever called the function. The manageLinks function always returns “false”. That’s its only purpose for the moment: it is its only line. All JavaScript lines that do something, such as the “return” line, end in a semicolon.

Objects and events

The final line overrides an event. The “onclick” event happens every time someone clicks something anywhere on the web page. The web page itself is an object. In JavaScript the web page is the “document”. We’re telling the document—the web page—that every time the visitor creates an “onclick” event (by clicking something) to call our manageLinks function.

So, what happens when the visitor clicks on a link on the web page? The link itself is an object, usually an HTMLAnchorElement (we’ll deal with that sort of thing later). Whatever object they clicked on generates an event, in this case an onclick, and starts bubbling that event up through the document. If the link is contained in a paragraph element, the onclick bubbles up through the paragraph. If the paragraph is contained in a body element, the onclick bubbles up through the body, and so on, until it hits the topmost parent of all of the elements in the web page, the document.

Once the onclick event hits the document, we’ve told document that all onclicks need to call manageLinks.

The manageLinks function always returns false. That tells the document not to do anything else. If we returned nothing, or if we returned true, the document would know to continue doing what it otherwise would have done. You can try that now. Change the ‘false’ to ‘true’, reload the web page, and try clicking some links. Now they’ll work.

BLOCK EXTERNAL LINKS

Blocking all links isn’t particularly useful. Eventually we don’t want to block any links, we want the browser to treat some links differently.