**Programming Overview: Terms and Concepts**

Up to this time, you have programmed solely in HTML and CSS, which are DECLARATIVE languages; they declare what something is (HTML) and what that something is supposed to look like (CSS). If you actually want something specific to happen on a web page (such as opening a new window or processing a form), you will need to use an IMPERATIVE language, a language that says "do this!", a language, in short, like JavaScript.

Simple activities in JavaScript, like opening a new window or creating custom status bar messages, require minimal programming expertise; scripts for these sorts of things can be had in any decent JavaScript "cookbook". The purpose of these lectures is to present an overall concept of programming in JavaScript which will give you a foundation, upon which you can build, to permit you to go beyond the simple scripts.

In JavaScript, a program, called a FUNCTION, is a collection of lines of text-based code which performs some activity.

JavaScript is an "event-driven" programming language, which means that functions in JavaScript are triggered by events. What is an EVENT? An event, in JavaScript, is any user interaction with the computer. Clicking the mouse is an event. Pressing a key on the computer keyboard is an event. Moving the mouse around on the screen is an event. Loading an HTML page in the browser is an event. Any action initiated by the user constitutes an event.

You already know what a RESOURCE is: a resource is any piece of digital media, whether text, sound, graphic, animation, form element, or CSS document, among others. Think beyond your current definition of resource. Resources can include individual pieces of code, functions, windows, dialog boxes, scrollbars, interface elements, etc. Almost anything in the computer, anything digital, can potentially be a resource which you may then access and/or manipulate in JavaScript.

When you are programming in JavaScript, you will write FUNCTIONS which are triggered by user-initiated EVENTS; these functions will, in their turn, access and manipulate RESOURCES for various purposes. In other words, the user will interact with the computer, triggering a JavaScript program; that program will do something, then use resources to display something for the user to see, hear, or experience (this is called FEEDBACK).

You, as the programmer, are designing the feedback for the user, as well as figuring out how the program itself is supposed to work. You have to design every aspect of the user experience or USER INTERFACE, the user interface being that portion of the program which the user will interact with that includes both interface elements (such as buttons and text fields) as well as feedback (such as dialog boxes, screens, sounds, etc).

When you are programming in JavaScript, you need to ask yourself what YOU would expect if you were using this device. Would it beep? Would the buttons hilite as you clicked them, and would you want to see rollover effects (like active/inactive button states)? What kind of response would you expect from the computer once the program is finished executing? Would you want to see dialog boxes? Should the HTML page advance to a new page? Should a window open? And what would really IRRITATE you? What do you want to make sure DOESN'T happen? You, as the programmer, must define every aspect, every detail, of the UI (User Interface), including every aspect of how it behaves.

But remember that old adage, KISS: "Keep It Simple, Stupid!" Don't include flashy highlights and rollovers that distract from the functionality of the program. A good interface is almost invisible, assisting the user without being obtrusive.

Having said all of this, the user interface is, in some respects, merely the clothing on the living creature of your program. The real flesh and bone of a program lies in the calculations, the operations, the decisions being made by the program itself which actually do the task at hand. You can obsess about what your buttons are going to look like, or what your dialog boxes are going to say, until your hair falls out, but if you haven't got code inside of your program deciding something or solving something or calculating something or moving something around, you're not going to have anything useful in the end.

Imperative programming languages, such as JavaScript, C/C++, and Java, use something called LOGIC to make simple decisions and perform repetitive actions in a program. In addition, these languages use OPERATIONS to perform mathematical calculations, comparisons, transfers of data, and other activities within the computer itself. In combination, logic and operations do all of the actual work inside of a program.

How logic operates, and what sort of operations JavaScript is capable of, are topics for discussion in later sections. All that you have to remember right now is that computers are really very stupid creatures; they can't do anything for themselves unless you tell them how to do it, which means that you have to spell EVERYTHING out for them. In addition, the operations and decisions computers can make are really very simple ones compared to the ones humans can make; sophisticated programs create an ILLUSION of complex behavior by layering many simple operations together and performing them at unbelievably high speeds. At the core of any program, however, are basic logic structures and basic operations which we will introduce in these modules.

Computer programming is mostly comprised of problem solving: taking large complex problems, breaking them down into the smallest possible pieces, and building programs out of these simple bits. Whenever students ask me how to tackle a problem, I always respond that they should WRITE DOWN the things that they are sure about, even the stupidest-seeming things. Once you have written down things you KNOW must happen or problems you have already solved, it makes it much clearer what remains to be done. In addition, you must start breaking down your problems into the simplest bits that you can think of, until you have the entire problem broken down into single tasks which the computer can perform. I will attempt to explain a little more about this process in the next module.

**Summary**

In JavaScript, user-initiated EVENTS trigger JavaScript FUNCTIONS. These functions contain LOGIC which makes decisions or performs repetitions, using OPERATIONS to calculate, to compare, or to solve various problems. Once all logic and operations are finished, the function accesses and/or manipulates RESOURCES to generate user FEEDBACK based on the decisions and operations it has made. Obviously, this is an artificially simplified statement (since logic, operations, resources, and functions would actually be a part of ALL of these processes), but it gives you a GENERAL idea of the process involved in the execution of a JavaScript program.

You, as a programmer, will design every aspect of the JavaScript code itself, as well as the USER INTERFACE. You will be taking complex problems, breaking them down into single tasks which the computer can execute, and reassembling these simple bits into larger-scale programs which can then perform the complete task.