**Chapter 4. Understanding JavaScript**

WHAT YOU’LL LEARN IN THIS CHAPTER:

• What web scripting is and what it’s good for

• How scripting and programming are different (and similar)

• What JavaScript is and where it came from

• How to include JavaScript commands in a web page

• What JavaScript can do for your web pages

• Beginning and ending scripts

• Formatting JavaScript statements

• How a script can display a result

• Including a script within a web document

• Testing a script in your browser

• Modifying a script

• Dealing with errors in scripts

• Moving scripts into separate files

The World Wide Web (WWW) began as a text-only medium—the first browsers didn’t even support

images within web pages. The Web has come a long way since those early days, as today’s websites

include a wealth of visual and interactive features in addition to useful content: graphics, sounds,

animation, and video. Web scripting languages, such as JavaScript, are one of the easiest ways to

spice up a web page and to interact with users in new ways.

The first part of this chapter introduces the concept of Web scripting and the JavaScript language. As

the chapter moves ahead, you’ll learn how to include JavaScript commands directly in your HTML

documents and how your scripts will be executed when the page is viewed in a browser. You will

work with a simple script, edit it, and test it in your browser, all the while learning the basic tasks

involved in creating and using JavaScript scripts.

**Learning Web Scripting Basics**

In the world of science fiction movies (and many other movies that have no excuse), computers are

often seen obeying commands in English. Although this might indeed happen in the near future,

computers currently find it easier to understand languages such as BASIC, C, and Java.

You already know how to use one type of computer language: HTML. You use HTML tags to

describe how you want your document formatted, and the browser obeys your commands and shows

the formatted document to the user. But because HTML is a simple text markup language, it can’t

respond to the user, make decisions, or automate repetitive tasks. Interactive tasks such as these

require a more sophisticated language: a programming language, or a *scripting* language.

Although many programming languages are complex, scripting languages are generally simple. They

have a simple syntax, can perform tasks with a minimum of commands, and are easy to learn. Web

scripting languages enable you to combine scripting with HTML to create interactive web pages.

**Scripts and Programs**

A movie or a play follows a script—a list of actions (or lines) for the actors to perform. A web script

provides the same type of instructions for the web browser. A script in JavaScript can range from a

single line to a full-scale application. (In either case, JavaScript scripts usually run within a

browser.)

**Note**

Is JavaScript a scripting language or a programming language? It depends on who you ask. We’ll refer

to scripting throughout this book, but feel free to include JavaScript programming on your résumé after

you’ve finished this book.

Some programming languages must be *compiled*, or translated, into machine code before they can be

executed. JavaScript, on the other hand, is an *interpreted* language: The browser executes each line

of script as it comes to it.

There is one main advantage to interpreted languages: Writing or changing a script is very simple.

Changing a JavaScript script is as easy as changing a typical HTML document, and the change is

enacted as soon as you reload the document in the browser.

**Note**

Interpreted languages have their disadvantages—they can’t execute quickly, so they’re not ideally suited

for complicated work, such as graphics. Also, they require the interpreter (in JavaScript’s case, usually

a browser) to work.