Chapter 1
All You Ever Wanted to Know about JavaScript (But Were Afraid to Ask!)

In This Chapter

- Understanding a working definition of JavaScript
- Dispelling common JavaScript misconceptions
- Getting started with JavaScript tools
- Finding online information

Ever surfed to a Web site that incorporates really cool features, such as

- Images that change when you move your mouse over them?
- Slide-show animations?
- Input forms with pop-up messages that help you fill in fields correctly?
- Customized messages that welcome repeat visitors?

Using JavaScript and the book you're reading right now, you can create all these effects and many more! The web page in Figure 1-1 shows you an example of the kinds of things you can look forward to creating for your own site.

JavaScript lets you add interactive features to your Web site quickly and easily.

A lot has changed since the previous edition of JavaScript For Dummies®. For one thing, JavaScript has evolved from a

relatively basic scripting language to a full-blown programming language in its own right.

The good news is that you can use JavaScript to create even more breath- takingly cool Web sites than ever before. The bad news is that the JavaScript language itself is a bit more complex than it was in its earlier incarnations -- but that's where this new, improved, better-tasting edition of JavaScript For Dummies® comes in! Even if you're not a crackerjack programmer, you can use the techniques and sample scripts in this book to create interactive, "intelligent" Web pages bursting with animated effects.

Before you hit the JavaScript code slopes, though, you might want to take a minute to familiarize yourself with the basics presented in this chapter. That way you'll have all the background you need to get started using JavaScript as quickly as possible!

What Is JavaScript? (Hint: It's Not the Same Thing as Java!)

JavaScript is a scripting language you can use -- in conjunction with HTML -- to create interactive Web pages. A scripting language is a programming language designed to give folks easy access to pre-built components. In the case of JavaScript, those pre-built components are the building blocks that make up a Web page (links, images, plug-ins, HTML form elements, browser configuration details, and so on).

You don't need to know anything about HTML to put this book to good use; I explain all the HTML you need to know to create and use the JavaScript examples you see in this book. If you're interested in learning more about HTML, I suggest checking out a good book devoted to the subject. One to try is HTML For Dummies®, 3rd Edition, by Ed Tittel and Steve James (published by IDG Books Worldwide, Inc.).

It's easy! (sort of)

JavaScript has a reputation of being easy to use because:

- The bulk of the document object model (the portion of the language that defines what kind of components, or objects, you can manipulate in JavaScript) is pretty straightforward.

For example, if you want to trigger some kind of event when a push button is clicked, you access the `onClick` event handler associated with the `button` object; if you want to trigger an event when an HTML form is submitted, you access the `onSubmit` event handler associated with the `form` object. (You become familiar with the JavaScript object model in this book by examining and experimenting with working scripts. You can also check out Appendix C, which lists all of the objects that make up the document object model.)

- When you load a cool Web page into your browser and wonder how the author created the effect in JavaScript, ninety-nine times out of a hundred all you have to do to

satisfy your curiosity is click to view the source code (View*Page Source in Navigator, View*Source in Internet Explorer). (Chapter 3 describes the .js files that are responsible for that hundredth time.) This source code free-for-all, which is simply impossible with compiled programming languages such as Java, helps you decipher JavaScript programming by example.

Having said all that, becoming proficient in JavaScript isn't exactly a no-brainer, either. One of the biggest factors contributing to the language's growing complexity is the fact that the two major JavaScript-supporting browsers on the market (Netscape Navigator and Microsoft Internet Explorer) implement JavaScript differently. Netscape supports JavaScript directly -- hardly a surprise, since Netscape was the one who came up with JavaScript in the first place! Internet Explorer, on the other hand, supports JavaScript indirectly by providing support for JScript, its very own JavaScript-compatible language. And despite strident claims by both Netscape and Microsoft that JavaScript and JScript, respectively, are "open, standardized scripting languages," neither company offers explicit, comprehensive, all-in-one-place details describing

- Precisely which version of JavaScript (or JScript) is implemented in each of their many interim browser releases (an interim browser release contains a decimal point; for example, Navigator 4.6 and Internet Explorer 5.5 are interim browser releases)

- Precisely what programming features are included in each version of JavaScript/JScript

- How each version of JavaScript compares to each version of JScript (as you see in Chapter 4, JavaScript and JScript differ substantially)

The upshot is that creating cross-browser, JavaScript-enabled Web pages now falls somewhere around 6 on a difficulty scale of 1 to 10 (1 being the easiest technology in the world to master and 10 being the hardest).

Fear not, however. Armed with an understanding of HTML and the tips and sample scripts you'll find in this book, you'll be a JavaScript jockey in no time flat!

What's in a name?

A long time ago, JavaScript was called LiveScript. In a classic "if you can't dazzle them with brilliance, baffle them with marketing" move, Netscape changed the name to take advantage of the burgeoning interest in Java (another programming language that Netscape partner Sun Microsystems was developing at the time). By all accounts, the strategy worked; unfortunately, many newbies still mistake JavaScript for Java, and vice versa.

Here's the scoop: Java is similar to JavaScript in that they're both object-based languages developed for the

Web. Without going into the nitty-gritty details of syntax, interpreters, variable typing, and just-in-time compilers, all you have to remember about the difference in usage between JavaScript and Java is this: on the gigantic client/server application that is the Web, JavaScript lets you access Web clients (otherwise known as browsers); Java lets you access Web servers.

This difference may seem esoteric, but it can be very useful in helping you determine which language to use to create the Web site of your dreams. If what you want to accomplish can be achieved inside the confines of a Web client (in other words, by interacting with HTML, browser plug-ins, and Java applets), JavaScript is your best bet. But if you want to do something fancier -- say, interact with a server-side database -- you'll need to look into Java or some other server-side alternative.

It's speedy!

Besides being relatively easy, JavaScript is also pretty speedy. Like most scripting languages, it's interpreted (as opposed to being compiled). When you program using a compiled language, such as C++, you must always reformat, or compile, your code file before you can run it. This interim step can take anywhere from a few seconds to several minutes or more.

The beauty of an interpreted language like JavaScript, on the other hand, is that when you make changes to your code -- in this case, to your JavaScript script -- you can test those changes immediately; you don't have to compile the script file first. Skipping the compile step saves a great deal of time during the debugging stage of Web page development.

Another great thing about using an interpreted language like JavaScript is that testing an interpreted script isn't an all-or-nothing proposition, the way it is with a compiled language. For example, if line 10 of a 20-line script contains a syntax error, the first half of your script still runs; you still get feedback immediately. The same error in a compiled program would prevent the program from running at all.

The downside of an interpreted language is that testing is on the honor system. Because there's no compiler to nag you, you might be tempted to leave your testing to the last minute or -- worse yet -- skip it altogether. But remember: whether the Web site you create is for business or pleasure, it's a reflection on you -- and testing is essential if you want to look your very best to potential customers, associates, and friends. (A few years ago visitors to your site might have overlooked a buggy script or two, but frankly, standards out there are much higher these days.) Fortunately, Chapter 17 is chock-full of helpful debugging tips to help make testing your JavaScript code as painless as possible.

Everybody's doing it! (Okay, almost everybody!)

Two generally available Web browsers currently support JavaScript: Microsoft's Internet Explorer and Netscape/AOL's

Navigator. (Beginning with version 4.0, Navigator became synonymous with Communicator, even though technically Netscape Communicator includes more components than just the Navigator Web browser.) Between them, these two browsers have virtually sewn up the browser market; almost everyone who surfs the Web is using one or the other (and thus has the ability to view and create JavaScript-enabled Web pages).

Web browsers aren't the only software tools that provide support for JavaScript, however; Netscape's Web servers support JavaScript, as well -- albeit a special server-side version.

This book focuses on client-side JavaScript, the most popular and widespread form by far. Netscape did develop a special version of JavaScript that runs on certain Web servers instead of in Web browsers, however. For more information on server-side JavaScript, visit

`http://developer.netscape.com/docs/manuals/ssjs/1_4/contents.htm`

JavaScript and HTML

You can think of JavaScript is an extension to HTML; an add-on, if you will.

Here's how it works. HTML tags create objects; JavaScript lets you manipulate those objects. For example, you use the HTML . . . tags to create a Web page document. Once that document is created, you use a special JavaScript construct called the `onLoad` event handler to trigger an action -- play a little welcoming tune, perhaps -- when the document is loaded into a Web browser. Examples of other HTML objects that you interact with using JavaScript include windows, text fields, images, and embedded Java applets.

Keep in mind that most of the examples in these printed pages focus on the JavaScript portion of the code (naturally!). But, I include the HTML you need to create the examples on the CD-ROM, so you don't have sweat recreating the web pages from scratch!

Because Web pages are not made of HTML alone, however, JavaScript provides access to more than just HTML objects; it also provides access to browser- and platform-specific objects. Browser plug-ins (such as RealAudio and Adobe Acrobat), the name and version of a particular viewer's browser, and the current date are all examples of non-HTML objects you can work with using JavaScript.

Together, all the objects that make up a Web site -- HTML objects, browser- and platform-related objects, and special objects built right into the JavaScript language -- are known as the document object model (DOM for short).

JavaScript and Your Web Browser

You need to use Netscape Navigator 4.0 (or higher) or Microsoft Internet Explorer 5.0 (or higher) to use the latest enhancements in JavaScript demonstrated in this book. Not all browsers are created equal: Internet Explorer's support for JavaScript differs significantly from Navigator's. For details, check out Chapter 4, "JavaScript: The Non-Standard Standard."

Although you can create and view JavaScript scripts with an old version of one of these browsers, I recommend that you install the most current version of either Navigator or Internet Explorer. (What the heck -- they're both free!) The latest versions of each product boast the very latest JavaScript features and bug fixes; they're also the version you see described in the figures and examples in this book.

You can use another browser, such as Opera or America Online (or even another Internet protocol, such as FTP) to download the latest version of either Navigator or Internet Explorer and try it out. "What Do I Need To Get Started?" later in this chapter, is devoted to the ins and outs of obtaining and installing a JavaScript-enabled browser. For now, suffice it to say that:

- You need Navigator or Internet Explorer to work with JavaScript, which means that you have to be running one of the client platforms that supports these browsers. (Macintosh and Windows both support Navigator and Internet Explorer.)

- You need to be aware that people may use other, non-JavaScript-enabled browsers to view your Web pages -- or they may use JavaScript-enabled browsers with JavaScript support turned off. Either way, you have no way to guarantee that everyone who visits your page will be able to view your JavaScript handiwork. (Check out Chapter 4, "I Spy! Detecting Your Users' Browser Environment" for more information on this topic.)

What Can I Do with JavaScript That I Can't Do with HTML?

HTML lets you create static Web pages using tag building blocks, or objects; JavaScript lets you inspect and manipulate those objects to punch up static pages with intelligence, interactivity, and simple animations. (In other words, in order to use JavaScript, you need to use HTML.) Using JavaScript, you can even create a page that appears differently depending on who's viewing the page, what browser they're using to view it, and what time of day it is! Read on for details.

Add Intelligence To Your Pages

In software parlance, "intelligence" refers to any chunk of code that does anything snazzier than a simple input-output process. Accepting a user's zip code is an example of a simple input process; taking that zip code and using it to calculate sales tax is an example of software intelligence.

When it comes to adding intelligence to your Web pages with

JavaScript, the only limiting factor (besides your imagination, of course!) is this: you're only allowed to work with client-side objects. In other words, you can use JavaScript to inspect any object in the browser DOM and to perform any calculations you like -- but you can't use it to communicate with, say, an Oracle database sitting on a server somewhere.

Generally speaking, to access information outside the DOM -- whether it's on your user's hard drive or on a machine somewhere else on the Internet -- you need to use either Java or a CGI program. (CGI, or Common Gateway Interface, programs are stored on Web servers and are typically written in either Perl or C.)

In the Web's early years, the only way to add intelligence to Web pages was to create a CGI program -- and you can still do so, if you choose. However, there are two drawbacks associated with using CGI. One, creating a CGI program is trickier than crafting a JavaScript script (if only because you have to have access to a Web server to test your CGI program). And two, accessing a CGI program from a Web page requires a trip to the server -- a potential time suck (and an unnecessary one, if you can perform the same processing on the client using JavaScript).

Interact with users

The most common way to perk up your pages with JavaScript is to make them interactive: in other words, to do something useful when a user clicks a mouse or types a word. For example, you can use JavaScript to

- Load content into multiple frames when a user clicks a button
- Swap images when a user drags a mouse over a certain area of the screen (this effect is called a mouse rollover)
- Display helpful information when a user clicks or drags a mouse over a specified area of your page
- Inspect the data your users type in and pop up helpful suggestions if they've made an invalid entry
- Display a thank-you message after a user submits a form

In addition to user-initiated events such as clicking and dragging a mouse, JavaScript also recognizes automatic events -- for example, loading a Web page into a browser. (Check out Chapter 4 for details, including sample scripts that run in response to automatic events.)

Create cool animations

Many folks assume you need Java to create animations for the Web, but that's just not so. While JavaScript certainly won't be mistaken for the most efficient way to create high-density animations, you can use JavaScript with layers (sometimes known as dynamic HTML, or DHTML) to create a variety of really neat animated effects. As a matter of fact, using JavaScript is the

easiest way to implement common effects, such as rollovers, as you can see in Chapter 9.

Customize Your Pages

Everyone likes to feel special, and the folks who visit your Web site are no exception. Using JavaScript, you can tailor the way your pages look to different users based on criteria such as

- The specific kind and version of browser someone is using to view your page
- The current date or time
- Your users' behavior the last time they visited your page
- Your users' stated preferences
- Any other criterion you can imagine!

What Do I Need to Get Started?

I hope you're champing at the bit to get started on your first JavaScript-enabled Web page! First things first, though . . . You have an idea of what JavaScript can do for you, and by now you probably already have something specific in mind for your first attempt. Now's the time to dive into the preliminaries: what you need to get started and how to get what you need if you don't already have it. After you've dispensed with the setup, you can go on to the really fun stuff!

Hardware

For the purposes of this book, I assume that you're beginning your JavaScript adventure with either a personal computer or a Macintosh computer. Your machine (or box, to use the vernacular) should be a Pentium PC or better (unless it's a Power Mac) and should have at least 32MB of RAM and at least 25MB free disk space. If none of this makes sense, try asking your local hardware guru; every organization seems to have at least one. (I've found, through extensive trial and error, that most hardware gurus are fairly responsive to sugar-based snack foods.)

You also need hardware installed that lets you connect to the Internet. This hardware usually consists of a modem and a phone line. Depending on your computer, you may have an internal modem installed (many come complete with a built-in modem). If not, you can buy a modem at your local computer discount store. The differentiating factor among modems is line speed: the faster, the better. (Most computers these days come with a 56.6 Kbps model pre-installed, but 14.4 or 28.8 will work just fine.) If you don't already have a modem and need to purchase one, consider buying the fastest modem in your price range; you'll be very glad you did when you try to look at spiffy Web pages with many graphics, each of which takes a loooong time to load (because graphics files are typically so large).

Software

For the purposes of this book, I assume that you have either a Macintosh computer, or a personal computer loaded with Windows 95, Windows NT, Windows 98, Windows 2000, or Linux. (Currently, only Netscape Navigator is available for use with Linux.)

I also assume that you have some way to create text files. (Most operating systems come packaged with a variety of text editors and word processors, any of which will work just fine for creating JavaScript scripts.)

On the CD included with this book you'll find some great text editing utilities designed specifically for creating JavaScript files.

JavaScript-specific software

You also need a Web browser. Navigator (Netscape Communication's commercial Web browser) and Microsoft's Internet Explorer are the only generally available browsers that support JavaScript at the time of this writing. So, the first thing to do is to get a copy of Navigator or Internet Explorer.

(The examples you see in this book are demonstrated using both Netscape Navigator and Internet Explorer running on Windows 95.)

Most personal computers come with Internet Explorer already installed. To find out if this is the case for your particular computer, select Start*Programs and look for Internet Explorer.

Netscape Navigator

Netscape Navigator version 6.x bundles the Navigator browser with messaging, Web construction, and other Internet-related goodies.

You can download a copy by visiting the following site (which offers step-by-step installation instructions)

```
http://www.netscape.com/download/index.html
```

Of course, I'm assuming that you already have a Web browser installed or that you have access to FTP. (FTP is short for file transfer protocol, which is an Internet application that enables you to nab files from other people's machines.) If you prefer, you can purchase a copy of Navigator on CD-ROM for a nominal fee (details are available at Netscape's site).

Internet Explorer

If you're a Microsoft buff, you might want to download a copy of Internet Explorer. Download it for free (or order your copy on CD-ROM for a nominal fee) from the following site, which offers easy-to-follow installation instructions:

http://www.windows/ie/default.htm

Documentation

For the latest Netscape Navigator and Microsoft Internet Explorer documentation and technical support, respectively, check out the following URLs:

http://www.netscape.com

http://www.microsoft.com

To get a copy of the JavaScript Authoring Guide, the must-have tome from Netscape that explains JavaScript basics and language concepts (and includes an extensive reference section), visit the following Web page:

http://www.developer.netscape.com