Chapter 1

Taking the Web to the Next Level

In This Chapter

- Reviewing HTML, XHTML, and CSS
- Examining the role of JavaScript and AJAX
- Exploring what JavaScript and AJAX can add to Web pages
- Choosing an editor
- Building your browser toolkit

The Web is a pretty big deal. It's a lot of fun to build Web pages, and just about every business needs them. As the Web has grown and changed, the expectations of what a Web page is and does have also changed. If you already know HTML or XHTML, you know how to create Web documents — if you need a refresher, check out Bonus Chapter 1 on either the companion Web site at `www.dummies.com/go/javascriptandajaxfd` or my own site at `www.aharrisbooks.net/jad`.

As the Web has evolved so have the tools that are used to create Web pages and documents. JavaScript and AJAX are two powerful tools for creating dynamic Web documents. This chapter gets you started with a look at some of the primary technologies out there for building Web pages.

Building Something Cool

This book is about adding features to Web pages that you cannot do with simple HTML and CSS.

Make no mistake; we're talking about programming here — and programming is a little bit harder than plain old Web development. However, it's really worth it, as the example page in Figure 1-1 illustrates.

To keep this example simple, I'm using some external libraries. They are explained in Part IV of this book, but for now just appreciate that something exciting is happening here.

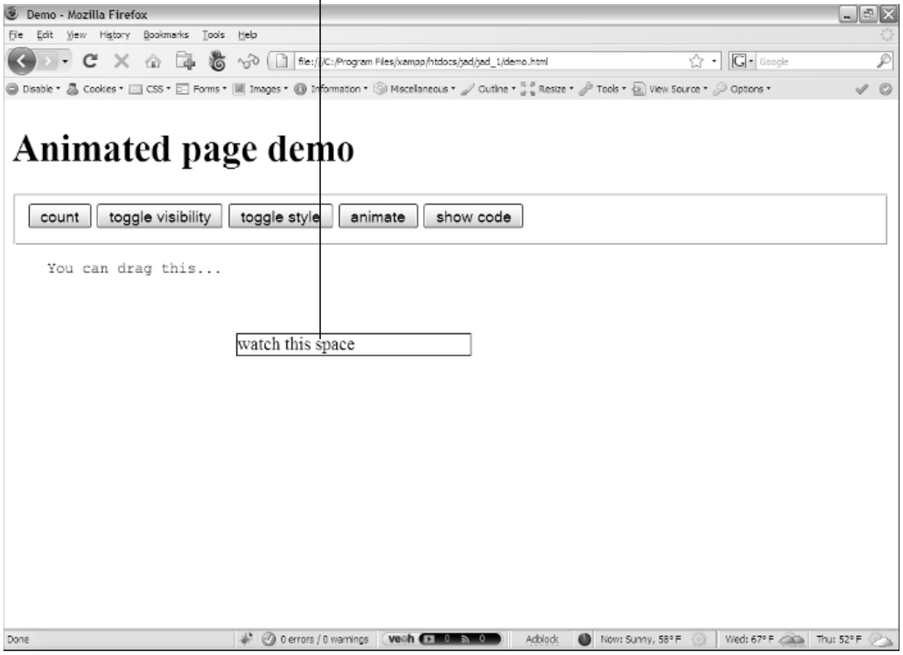


Figure 1-1: This page has some interesting features that would not be possible in basic HTML.

This program requires you to have an active Internet connection to work correctly. Check Chapter 10 to see some alternatives for connecting to external libraries.

TIP

If you want to see this page in action (and you really should), please go to the companion Web sites for this book: `www.aharrisbooks.net/jad` or `www.dummies.com/go/javascriptandajaxfd`. This program and every other program and example in the book are available at that site.

At first, the Web page looks pretty simple, but when you open it in your own browser (as you should) and begin playing with it, you'll soon discover that it packs a lot of surprises. This very simple page illustrates a lot of the reasons why you should learn JavaScript and AJAX.

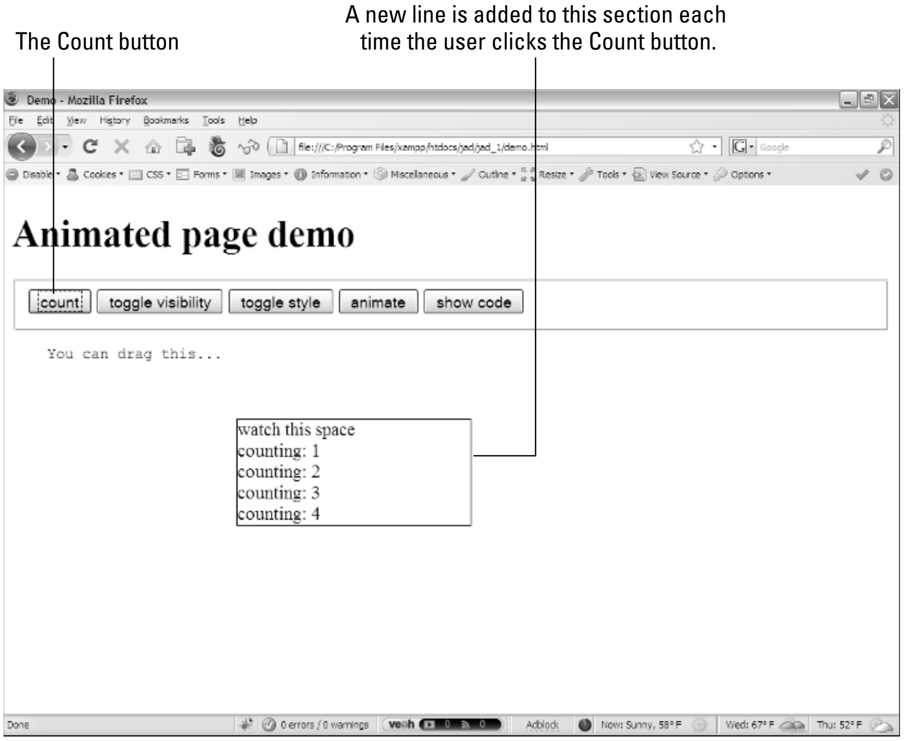
- **The buttons do something.** You might already have a handle on creating form elements (such as buttons and text fields) in plain HTML, but HTML can't do anything with the buttons and text fields; that's why you need a programming language.

  If you want something interesting to happen, you need a programming language. Each of these buttons uses JavaScript to do some interesting work, but the fact that the page is now interactive is a huge change. With JavaScript, you can build *applications*, not just pages.

- **The Count button dynamically changes the page.** When you click the Count button, new content is added to the page automatically. A program counts how many times the Count button is pressed and adds text to the "watch this space" section. As the user interacts with the page, the page has material that wasn't originally on the server.

  Of course, this example is simple, but you will be able to add any kind of text to any Web element dynamically. That's a very powerful capability. Figure 1-2 shows how the page looks after I click the Count button a few times.

- **The Toggle Visibility button makes things appear and disappear.** You can't really modify whether things appear or go away in HTML. You can do so in CSS to some level, but JavaScript gives you a much more powerful set of tools for changing what parts of the page are visible to the user at any time. Look at Figure 1-3 to see the page with the output segment hidden.



watch this space
counting: 1
counting: 2
counting: 3
counting: 4

Figure 1-2: The Count button changes the text in part of the page.

I clicked the Toggle Visibility button, and the output disappeared.

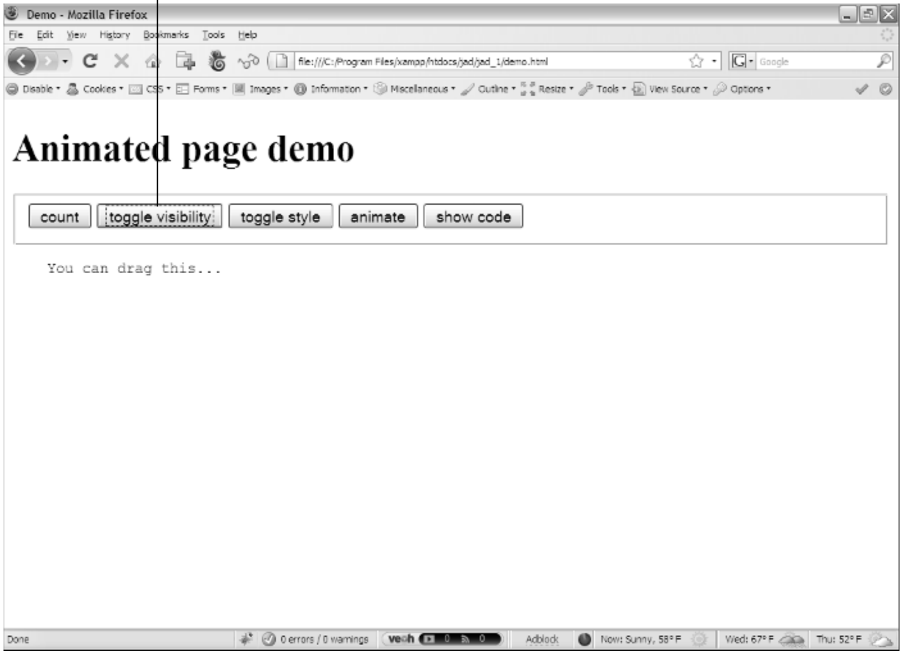


Figure 1-3: Click the Toggle Visibility button to make the panel with the output reappear.

- **The Toggle Style button instantly changes the appearance of part of the page.** You can use JavaScript to change the contents of any part of the page (that is, the HTML) — but you can *also* use JavaScript to modify the appearance (the CSS) in real time. In this example, I've created a special CSS class called `funky` that is added to or removed from the output box every time the user clicks the button. This approach works with any CSS class. (Amazing, huh?) Figure 1-4 shows the page with the `funky` class applied.

  TIP: If you need a refresher on CSS or XHTML, please look over the bonus chapters on the Web site: `www.dummies.com/go/javascriptandajaxfd` or `www.aharrisbooks.net/jad`.

  TIP: I've added callouts to some of the figures in this chapter to describe what's happening. The images in this book are not sufficient to understand what the page does. Find the program at `www.dummies.com/go/javascriptandajaxfd` or `www.aharrisbooks.net/jad` and look at it yourself.

- **The Animate button is even more fun.** The Animate button makes a series of gradual changes to the output box, changing its size, shape, and appearance over time. (You've really got to try it; a screen shot won't do it justice.)

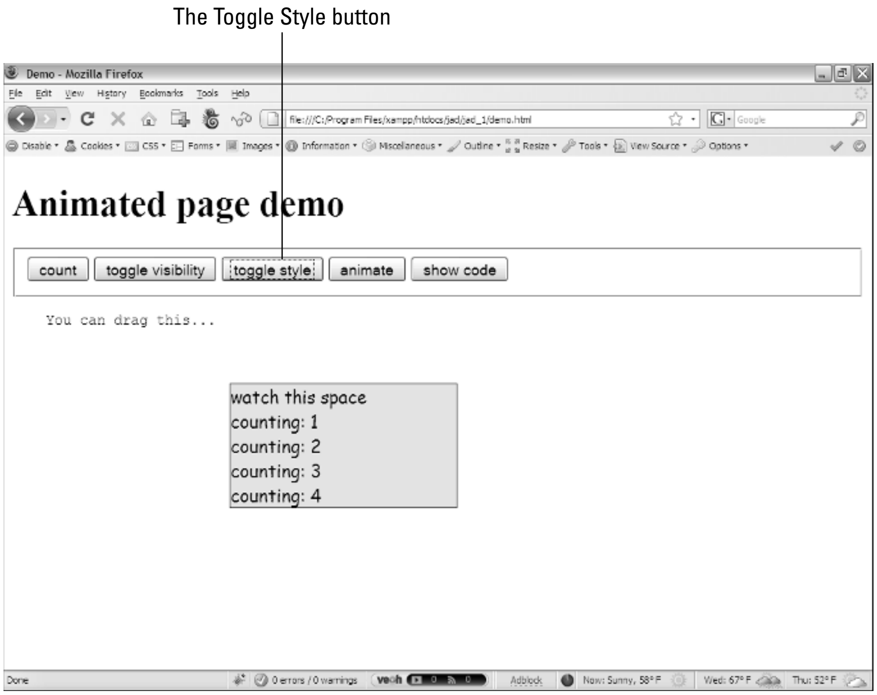


Figure 1-4: You can dynamically change the appearance of any page element.

- **The Show Code button brings in text from an external file.** This button uses a simple form of AJAX to load an external file into the page in real time. This is an easy way to build modular pages. In this case, I'm actually pulling in a copy of the JavaScript code so you can see how it works. Don't worry if you don't understand it yet. That's what this book is for! Figure 1-5 shows this operation in action.
- **Let the user drag the code.** The user can pick up the code and move it wherever she wants. This is another kind of functionality unheard of in ordinary Web pages.

TIP

No, you wouldn't normally display your code to users. I'm just illustrating that it's pretty easy to pull in an arbitrary text file from a server. Since you are a programmer now, I chose to show you a preview of the code as the text file I brought in.

These capabilities are profound, and they're just the beginning. Learning to program transforms your Web pages from static documents to living applications that can interact with your users in new and exciting ways.

TIP

Feel free to look over the code for this project. It's actually in three files: `demo.html`, `demo.css`, and `demo.js`. All can be found in the Chapter 1 folder of the book's companion Web site. Use View Source when the page is displayed in your browser to see the source code of the HTML file.

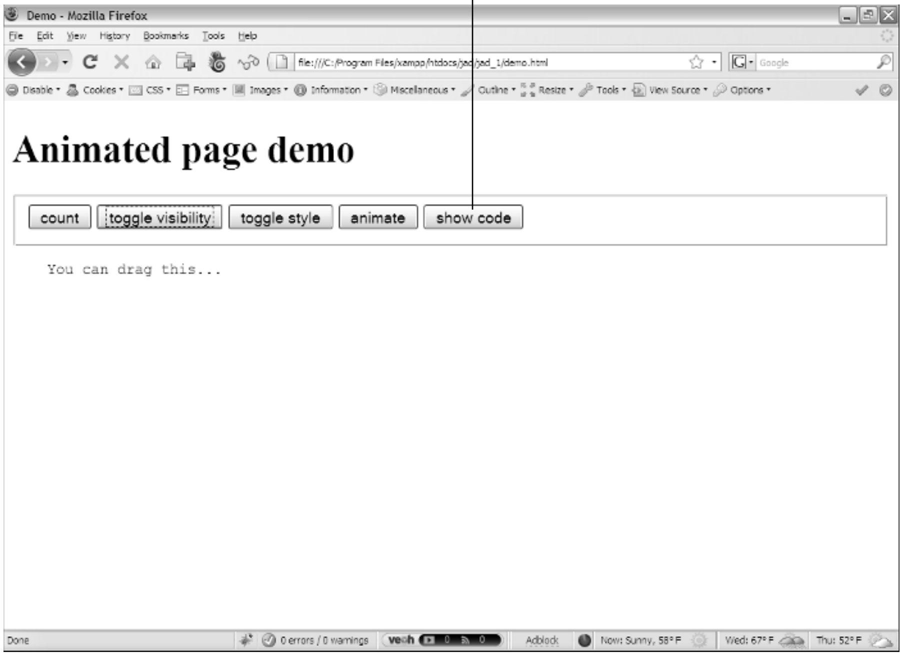


Figure 1-5: The Show Code button pulls code from another file and shows it on-screen.

I've added plenty of comments in the code to help you see what's going on, but it's okay if you don't have all the intricacies nailed down yet. Think of this as a preview of things you'll get to do in this book.

Getting Started

Making your pages do all this stuff looks like fun, and it is. There's a lot to get under your belt, but don't panic; I take you through everything. The first step is to review the core technologies that JavaScript and AJAX are based on, and see how they're related to some other (more advanced) technologies you'll eventually need.

Overview of the Core Technologies

Powerful as they are, JavaScript and AJAX do not stand on their own. They only have meaning in the context of Web pages, so they rely on various Web technologies. If you want to build a JavaScript application, you'll need several other technologies, too:

- **HTML:** HTML (HyperText Markup Language) is the basic markup language that describes Web pages. It's a relatively simple technique for building Web sites that requires nothing but a plain text editor.
- **XHTML:** XHTML is often considered the successor to HTML. Because it doesn't allow certain kinds of tags, XHTML is actually a smaller language that's a bit easier to use. Typically XHTML pages are more dependent on CSS than HTML, as many of the HTML tags are replaced with CSS tools.
- **CSS:** CSS (Cascading Style Sheets) is a way to add specific style information to an HTML or XHTML page. HTML and XHTML provide the general framework, and CSS describes the color and layout.
- **JavaScript:** JavaScript is a programming language embedded in all modern Web browsers. It's specially designed to interact with Web pages; you can use it to extract information from parts of a page, and to manipulate the page in real time.
- **AJAX:** (Asynchronous JavaScript And XML) is a technique that allows JavaScript to communicate more directly with the Web server. It creates an interesting new relationship between the Web browser and the Web server. About half of this book is dedicated to AJAX.
- **PHP:** (PHP Hypertext Preprocessor) is one of several important languages for working on a Web server. Although it's not a primary focus of this book, the PHP language can do things that JavaScript cannot do. AJAX is frequently used to connect JavaScript applications to PHP programs. You get a brief introduction to PHP in Chapter 14.
- **Java:** Java is a language that's entirely different from JavaScript (despite the similar names). Although Java is useful on both the client and server sides of the client-server relationship, it's not a primary focus of this book.

TIP

When you're looking for online help about JavaScript, be sure that you talk to Java*Script* experts and not *Java* programmers. Although the languages have similar names, they're entirely different languages. Java programmers love to act superior, and they'll give you grief if you ask a JavaScript question in a Java forum. If in doubt, ask on my Web site (`www.aharrisbooks.net`). I can help you with either language, and I won't mind (or bug you about it) if you're a little confused.

Choosing your computer

Of course, you'll need a computer. Fortunately, it doesn't have to be anything special. Any computer you can use to view Web pages can also be used to create them. Any of the major operating systems (Windows, Mac, and Linux) is perfectly fine. I do most of my work on a combination of Linux (Fedora Core) and Windows XP, but all the programs in the book will work exactly the same on any reasonably modern computer.

At some point you'll want your Web pages to be available on the Internet. Although you can install a server on your home computer, it's usually better to use an online hosting service. You can often get very good online hosting very cheaply or even free. If you want to have a specific name attached to your Web site (such as `www.mySite.com`), then you'll need to pay about $10 a year to register the domain. Hosting services frequently use Linux, but you'll probably use an online interface that hides all the details from you.

The right tools make any job easier, but for Web development, many of the really great software tools are available entirely free of charge. Because these tools are open source (available with a license that encourages distribution), they are entirely legal to use without paying for them, unlike commercial programs obtained using illicit methods.

You can do basic Web development on any computer with a text editor and browser. As your Web-tweaking skills get more sophisticated, you might want more powerful tools. Read on to see some great tools that cost absolutely nothing.

Picking an Editor

Web pages, JavaScript, HTML, and CSS are all ultimately forms of text. You don't really need any particular program to write them. Still, having exactly the right tool can make your life a lot easier. Since you're going to spend a lot of time with your Web tools, you should be aware of your options.

Avoiding the problem tools

Using the wrong tool for the job can really make your life difficult. Here are a few tools that don't really stand up to the job of Web development:

- **Microsoft Word:** Word processors are great (I'm using one to write this book), but they aren't really designed for creating Web pages. Word (and all other word processors) store lots of information in their files besides plain text. All the formatting stuff is great for non-Web documents, but HTML and CSS have their own ways of managing this data, and the other stuff gets in the way. Even the Save as HTML command is problematic. Although it stores the page in a form of HTML, Word's formatting is extremely clunky and difficult to work with. The resulting pages will not be suitable for adapting to JavaScript.
- **Notepad:** This is the classic tool built into most versions of Windows. It saves pages in plain text, so it's better than Word for Web development, but Notepad is too simplistic for any sort of serious work. It lacks such

basic features as line numbers — and it can't handle multiple documents at once. You'll quickly outgrow Notepad as a Web-development tool.

- **TextEdit:** The default text editor on the Mac is a very powerful tool, but it's more like a word processor than what I'd call a true text editor. When you save an HTML file in TextEdit, it's usually not stored the way you need it to: Rather than seeing the results of the code, you'll see the code itself. If you want to use TextEdit for HTML or JavaScript, make sure you choose Format⇨Make Plain Text before saving your file.
- **Graphics editors:** Some high-end graphics editors like Adobe Photoshop, Adobe Fireworks, and Gimp also have the ability to export to HTML, but the code they produce is not easy to work with. It's really better to use these programs to edit your graphics and use a dedicated text editor to handle your code.

Using a WYSIWYG editor

The promise of WYSIWYG ("what you see is what you get") editing is very alluring. Word-processing programs have had this capability for years. As you edit a document on-screen, you can see in real time exactly how it will look on paper. A number of tools promise this kind of functionality for Web pages: Adobe Dreamweaver is the most popular, followed by Microsoft FrontPage and its replacement ExpressionWeb. Although these tools are popular for traditional Web development, they have some drawbacks when it comes to the kind of interactive work we do in this book:

- **WYSIWYG is a lie.** The whole assumption of WYSIWYG works fine when the output is *a paper document printed on a printer.* You can predict how the output will work. Web pages are different, because the output shows up on a display that belongs to somebody else. You don't know what size it will be, what colors it will support, or what fonts are installed. You also don't know which browser the user will be viewing pages with, which can make a major difference in the output of the page.
- **The editor hides details you need.** A visual editor tries to protect you from some of the details of Web development. That's fine at first, but at some point you'll need that level of control. Most professionals who use Dreamweaver spend most of their time in Code view, ignoring the advantages of a visual editor. Why pay for features you're going to ignore?
- **Visual editors assume static documents.** A visual editor is based on the idea that a Web page is an ordinary document. The kinds of pages we build in this book are much more than that. You will (for example) be writing code that creates and modifies Web documents on the fly. You need to know how to build Web documents by hand so you can write code that builds them and changes them dynamically.

Introducing programmer's editors

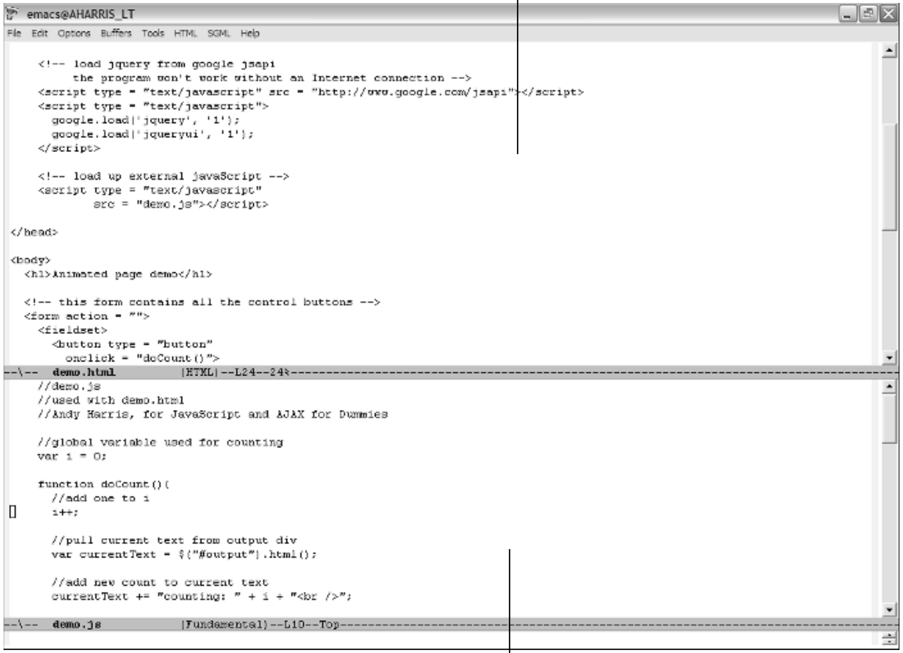
A number of specialty editors have propped up which seek to fill the gap between plain-text editors and the WYSIWYG tools. These editors write in plain text, but they have additional features for programmers, including:

- **Awareness of languages:** Programmer's editors often know what language you're writing in and can adapt, helping you whether you're writing HTML, JavaScript, or CSS code. Most general-purpose programmer's editors can handle all these languages natively, and often can help with many more languages.
- **Syntax highlighting:** Various elements are colored in different ways so you can see what is in plain text, what is part of an HTML tag, and so on. This simple feature can make it much easier to find problems like missing quotes, and to see the general structure of your page quickly.
- **Syntax support:** Programmer's editors often provide some sort of help for remembering the syntax of your language. This boost can be in the form of buttons and macros for handling common code, pre-written templates for standard layouts and patterns, and syntax completion (which looks at what you're typing and suggests completions based on the current language you're using).
- **Multiple document support:** Advanced Web applications often involve editing several different documents at once. You might have a dozen Web pages with a few CSS style sheets and an external JavaScript file or two. A programmer's editor allows you to view and edit all these files simultaneously. Many also allow you to generate a *project file* so you can save all the related files automatically and load them in one batch.
- **Macro tools:** Programming often requires repetitive typing tasks. Having a feature that records and plays back sequences of keystrokes as *macros* (short automated operations) can be incredibly helpful.
- **Debugging and preview support:** Most programmer's editors have a tool for previewing your code in a browser (or sometimes directly in the editor). The editors also often have tools for predicting certain errors, or responding to errors when they occur. At a minimum, you need the capability to jump directly to a particular line or section of your code.
- **Indentation support:** Most programmers use indentation as a powerful tool to help them understand the structure of the Web documents they're building. A good editor can assist you with this indentation and also help you recognize when you've made mistakes in the structure of your document.

Getting familiar with some important editors

A couple of multi-purpose programmer's editors immediately come to mind. You should consider investigating one or more of these free programs:

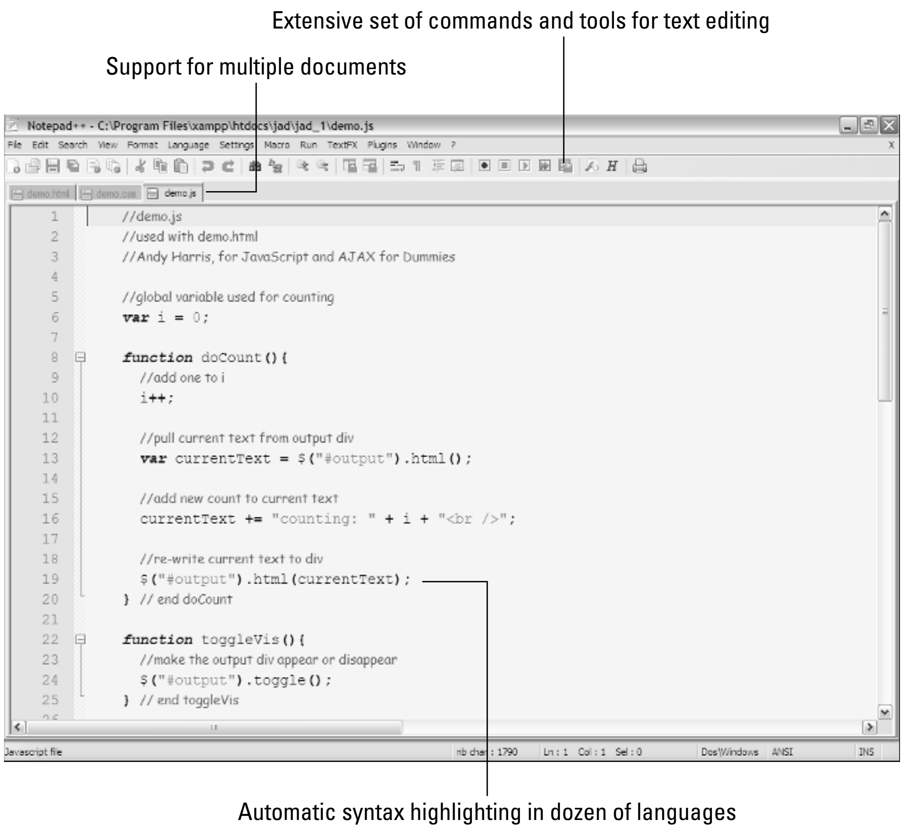
- **vi and emacs:** These are the granddaddies of all text editors. Both are very common on Unix/Linux environments. They are also available for Windows and Mac. Though extremely capable editors, vi and emacs were developed at a time when modern ideas about usability weren't practical. If you already know how to use one of these tools, by all means investigate a modern variant. (Frankly, I still use emacs as my primary text editor, though I don't know if I'd learn it today with all the easier options out there.) Figure 1-6 shows a Web page being edited with emacs.


Clean interface without buttons or gadgets
emacs@AHARRIS_LT
File Edit Options Buffers Tools HTML SGML Help
You can have many files open at once or look at two spots in the same file.

```
    <!-- load jquery from google jsapi
         the program won't work without an Internet connection -->
    <script type = "text/javascript" src = "http://www.google.com/jsapi"></script>
    <script type = "text/javascript">
      google.load('jquery', '1');
      google.load('jqueryui', '1');
    </script>

    <!-- load up external javaScript -->
    <script type = "text/javascript"
            src = "demo.js"></script>

</head>

<body>
  <h1>Animated page demo</h1>

  <!-- this form contains all the control buttons -->
  <form action = "">
    <fieldset>
      <button type = "button"
        onclick = "doCount()">
--\--  demo.html         |HTML|--L24--24%------------------------------------------
    //demo.js
    //used with demo.html
    //Andy Harris, for JavaScript and AJAX for Dummies

    //global variable used for counting
    var i = 0;

    function doCount(){
      //add one to i
      i++;

      //pull current text from output div
      var currentText = $("#output").html();

      //add new count to current text
      currentText += "counting: " + i + "<br />";
--\--  demo.js           |Fundamental)--L10--Top------------------------------------
```

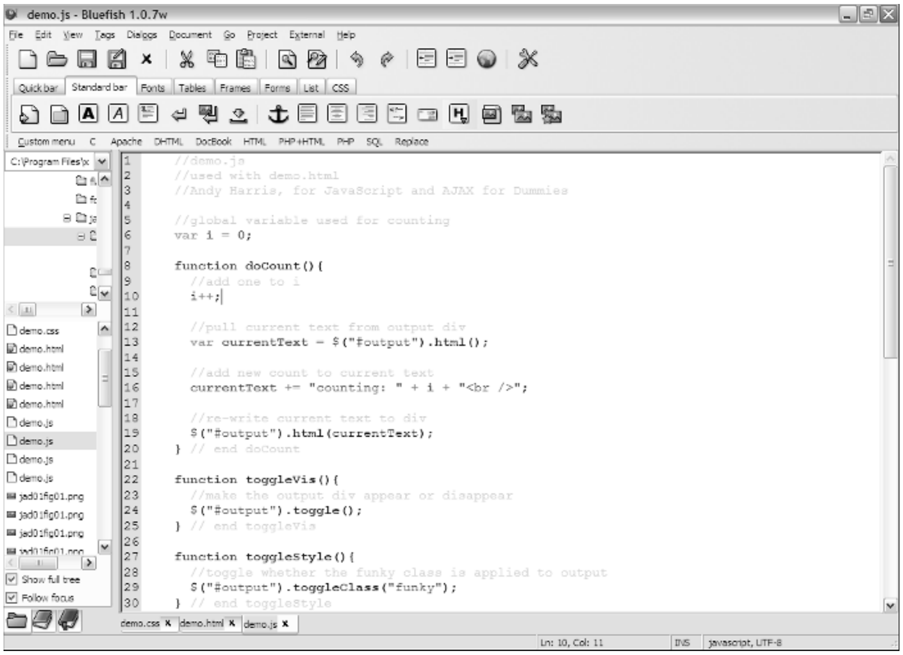
Figure 1-6: Emacs isn't pretty, but it's very powerful. Use it for extra geek points.

- **notepad++:** This is what Notepad for Windows should be. It starts with the speed and simplicity of Windows Notepad, but adds tons of features for programmers. I especially like the built-in support for page validation. This is one of the few programs to earn a permanent shortcut on my desktop. Unfortunately, it's only for Windows. Figure 1-7 shows the same page being edited in notepad++.
- **Bluefish:** The Bluefish text editor is rapidly becoming a favorite tool for Web developers. It's quick and powerful, and it has plenty of great features for Web developers. One especially powerful tool is the CSS generator, which helps you develop style sheets with a menu system so you don't have to memorize any syntax. It also has a great generator for default templates, which makes XHTML-strict Web pages much easier to build. Bluefish is available for all major platforms (for the Windows version, you'll also need to install the free GTK library). You can see Bluefish running in Figure 1-8.


Extensive set of commands and tools for text editing
Support for multiple documents
Notepad++ - C:\Program Files\xampp\htdocs\jad\jad_1\demo.js
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
demo.html demo.css demo.js
Javascript file nb char : 1790 Ln : 1 Col : 1 Sel : 0 Dos\Windows ANSI INS
Automatic syntax highlighting in dozen of languages

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for Dummies

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;

  //pull current text from output div
  var currentText = $("#output").html();

  //add new count to current text
  currentText += "counting: " + i + "<br />";

  //re-write current text to div
  $("#output").html(currentText);
} // end doCount

function toggleVis(){
  //make the output div appear or disappear
  $("#output").toggle();
} // end toggleVis
```

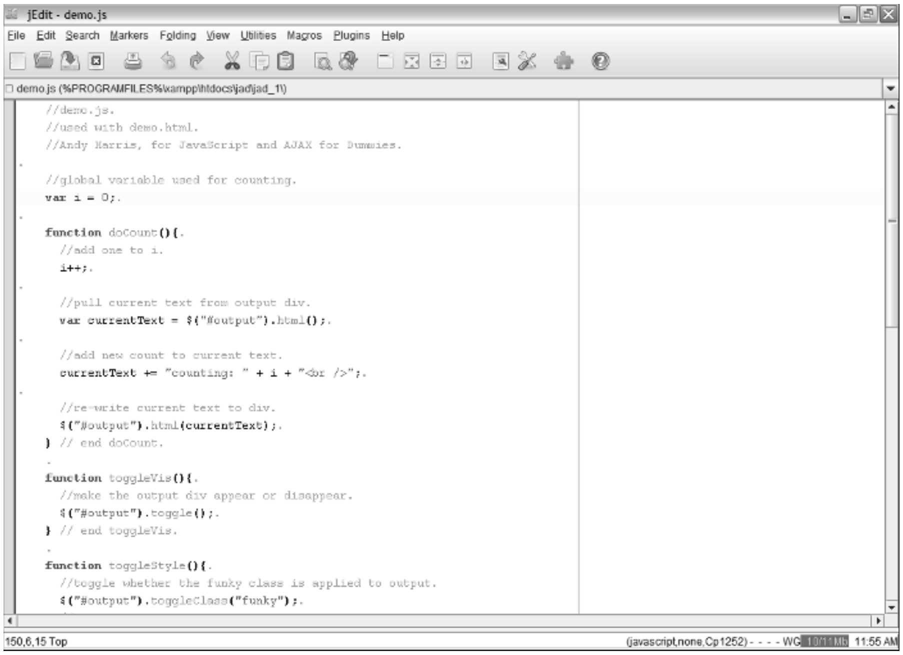
Figure 1-7: You'll find notepad++ a very powerful alternative to Notepad.


demo.js - Bluefish 1.0.7w
File Edit View Tags Dialogs Document Go Project External Help
Quick bar Standard bar Fonts Tables Frames Forms List CSS
Custom menu C Apache DHTML DocBook HTML PHP+HTML PHP SQL Replace
demo.css demo.html demo.js
Show full tree
Follow focus
Ln: 10, Col: 11 INS javascript, UTF-8

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for Dummies

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;

  //pull current text from output div
  var currentText = $("#output").html();

  //add new count to current text
  currentText += "counting: " + i + "<br />";

  //re-write current text to div
  $("#output").html(currentText);
} // end doCount

function toggleVis(){
  //make the output div appear or disappear
  $("#output").toggle();
} // end toggleVis

function toggleStyle(){
  //toggle whether the funky class is applied to output
  $("#output").toggleClass("funky");
} // end toggleStyle
```

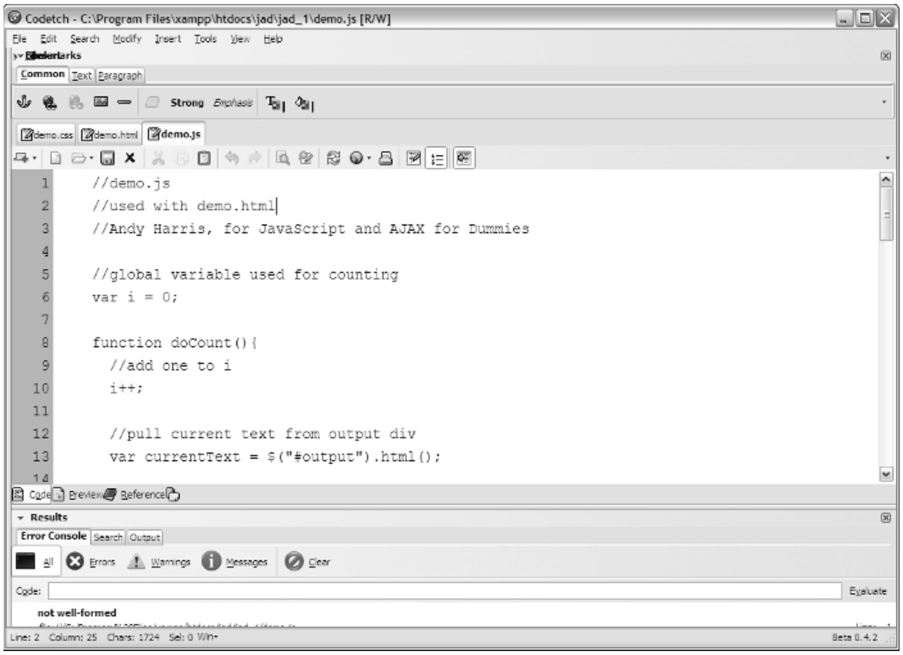
Figure 1-8: Bluefish is a very nice editor for XHTML and JavaScript.

- **jEdit:** This powerful editor is written in Java, so it is available on virtually every platform. It is a very powerful editor in its basic format, but its best feature is the extensive plugin library that allows you to customize it to your own needs. If you install the free XML library, jEdit has incredible support for HTML and XHTML. Figure 1-9 shows the sample program being edited in jEdit.
- **codetch:** This editor is unique because rather than being a standalone editor, it is actually an extension for the popular Firefox browser. It has most of the same features as the other editors, with the convenience of being already a part of your browser. It is not quite as configurable as some of the other tools, but it's still extremely handy. You can see codetch in action in Figure 1-10.


jEdit - demo.js
File Edit Search Markers Folding View Utilities Macros Plugins Help
demo.js (%PROGRAMFILES%\xampp\htdocs\jad\jad_1\)

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for Dummies

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;

  //pull current text from output div
  var currentText = $("#output").html();

  //add new count to current text
  currentText += "counting: " + i + "<br />";

  //re-write current text to div
  $("#output").html(currentText);
} // end doCount

function toggleVis(){
  //make the output div appear or disappear
  $("#output").toggle();
} // end toggleVis

function toggleStyle(){
  //toggle whether the funky class is applied to output
  $("#output").toggleClass("funky");
```

Figure 1-9: jEdit is a fast and capable editor written in Java.


Codetch - C:\Program Files\xampp\htdocs\jad\jad_1\demo.js [R/W]
File Edit Search Modify Insert Tools View Help
Common Text Paragraph
Strong Emphasis
demo.css demo.html demo.js

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for Dummies

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;

  //pull current text from output div
  var currentText = $("#output").html();
```

Code Preview Reference
Results
Error Console Search Output
All Errors Warnings Messages Clear
Code: Evaluate
not well-formed
Line: 2 Column: 25 Chars: 1724 Sel: 0 Win
Beta 0.4.2

Figure 1-10: The codetch plugin for Firefox is a complete Web editor.

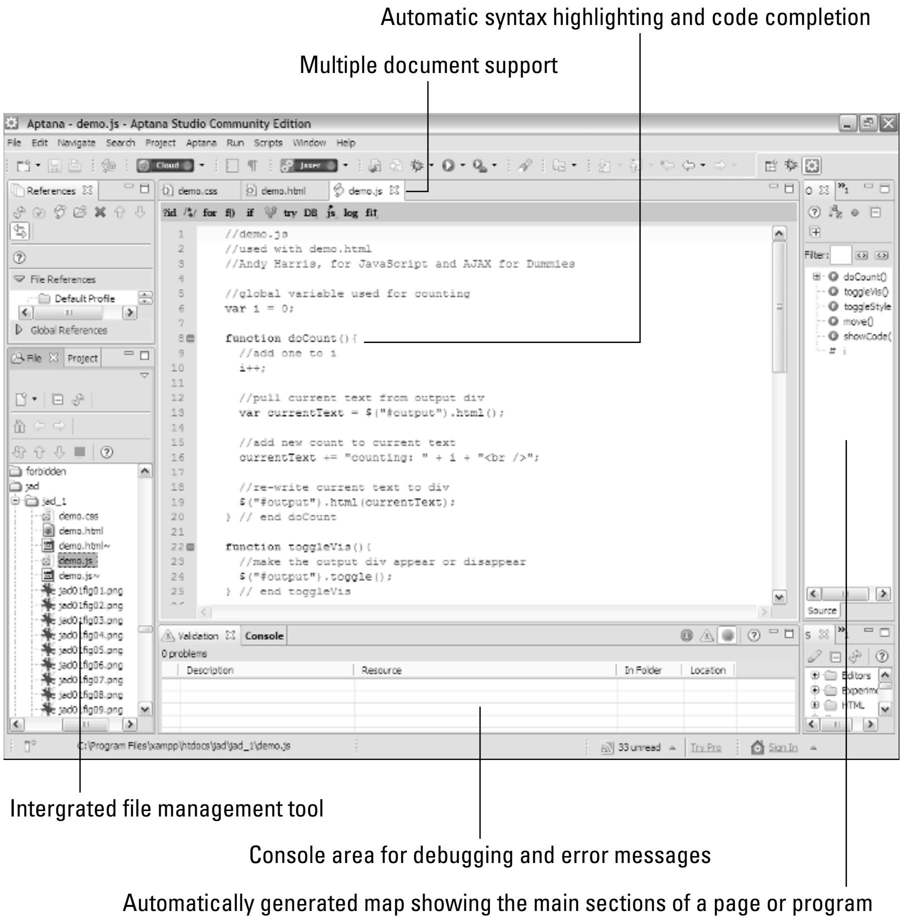
Introducing Aptana

One particular programmer's editor has really taken over the Web development world in recent years. Aptana is a full-featured programmer's editor based on the powerful and popular Eclipse editor for Java programming. Aptana has a lot to recommend:

- **Extensive built-in support for Web languages:** Aptana comes out of the box with support for HTML/XHTML, CSS, JavaScript, and AJAX.
- **Syntax highlighting:** Most programmer's editors have syntax highlighting, but Aptana is especially capable in this area. Sometimes you'll have the same physical document with three or more different languages active, and Aptana can usually sense by context whether you're writing CSS, XHTML, or JavaScript code.
- **Code completion:** This is one of Aptana's most impressive features. When you start writing a line of code, Aptana will pop up a menu of suggestions. This helps you avoid mistakes, so you don't have to memorize all the various CSS attributes and JavaScript commands exactly.
- **Error detection:** Aptana can look over your document as you create it and highlight some areas in real time. This feature can help you write better code, and can also help hone your skills at writing code.
- **AJAX support:** AJAX is a relatively new technology, and most editors do not directly support it. Aptana has a number of features that help you with AJAX, including built-in support of all the major AJAX libraries.

Aptana is completely free. I've placed a link to Aptana (and indeed all the tools mentioned here) on the Web site for this book. You can see Aptana in action in Figure 1-11.

My personal setup varies from machine to machine, but generally I use Aptana for my heavy programming, with notepad++ as a quick editor on Windows, and emacs as my primary basic text editor on Linux or Mac machines. Of course, you'll develop your own preferences as you go. All these editors are free and available at `www.aharrisbooks.net/jad`, so they're worthy of some experimentation.


Automatic syntax highlighting and code completion
Multiple document support
Aptana - demo.js - Aptana Studio Community Edition
demo.css
demo.html
demo.js
Validation
Console
0 problems
Description
Resource
In Folder
Location
Intergrated file management tool
Console area for debugging and error messages
Automatically generated map showing the main sections of a page or program

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for Dummies

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;

  //pull current text from output div
  var currentText = $("#output").html();

  //add new count to current text
  currentText += "counting: " + i + "<br />";

  //re-write current text to div
  $("#output").html(currentText);
} // end doCount

function toggleVis(){
  //make the output div appear or disappear
  $("#output").toggle();
} // end toggleVis
```

Figure 1-11: Aptana might be the best Web editor available at any price.

Creating Your Browser Collection

Web pages live within the context of Web browsers. Each browser interprets HTML and CSS a bit differently, and the differences are magnified when you start talking about JavaScript and AJAX. Subtle (and sometimes not-so-subtle) differences in the way browsers support your code can be very important.

Setting the standard

Every Web browser has its own particular way of displaying Web pages. Although those ways are pretty similar, the differences can sometimes be a problem. Worse, as you begin to write JavaScript code, you'll find that each browser has its own interpretation of the code. That can be a real mess.

Fortunately, there's been a big push toward standardization in recent years. The various browser developers have been getting together and agreeing to various standards set forth by a centralized team called the *World Wide Web Consortium* (W3C). When a browser implements JavaScript, it now agrees (theoretically, at least) to adhere to a set of standards for behavior. As long as your code follows the same standards, you can expect everything to work pretty well. (Most of the time, anyway.)

TIP

In this book, I adhere to accepted JavaScript standards as practiced by most developers. All the code in this book is tested on IE7 for Windows, Firefox 3 for Windows, and Firefox 3 for Linux. Any time the code is likely to cause particular browser problems, I try to point out the specific issues.

Picking a browser or two

Here are a few browsers you should be aware of:

- **Legacy browsers:** You'll find a lot of older browsers still being used on the Internet. Some people have continued to stick with whatever browser was on their machine when they got it, and haven't upgraded in years. The browsers earlier than IE6 or Firefox are a particular problem, because support for Web standards *and* for JavaScript was very uneven in the early days of the Web. For the most part, this book assumes that your users will be using at least a somewhat modern browser.

  REMEMBER

  AJAX in particular won't work on really old browsers.
- **Microsoft Internet Explorer 6:** This is a very common browser, still in popular use. At one point it was the dominant browser on the Internet, but it has fallen from favor in recent years, being replaced by newer offerings from Microsoft as well as competitors like Firefox and Opera. This browser is well-known for a number of "features" that don't comply with community standards. Its use is declining, but as of this writing, you still have to consider supporting it; a lot of users still have it.
- **Firefox:** The Firefox Web browser from Mozilla reopened the so-called "browser wars" by providing the first significant competition to Microsoft in many years. Firefox really opened eyes with its impressive features: tabbed browsing, improved security, and integrated searching. For developers, Firefox was among the first browsers to truly support Web standards in a serious way. Firefox is especially important to developers because of its extension architecture, which allows a programmer to turn Firefox into a high-powered development tool. Look at the next section of this chapter for suggestions on some great extensions to add to Firefox.

- **Microsoft Internet Explorer 7 and 8:** IE7 could be considered a tribute to Firefox, as it incorporates many of the same features. While this book was being written, IE8 came out, and added a few more improvements. Although the support for standards is not as complete in IE7 and IE8 as it is in some of the other current fleet of browsers, they are much better than in any earlier versions of IE.
- **Opera:** Opera is an important browser because it was one of the earliest browsers to actively support Web standards. It's very popular in certain circles, but has never gained widespread popularity. Since it supports Web standards, it will typically run any code written for a standards-compliant browser.
- **Safari:** Safari is the Web browser packaged with Mac OS. It is a very capable standards-compliant browser. There is now a Windows version available. The Web browser built into iPhones uses the same engine as Safari, so this is an important consideration if you're building applications for mobile devices.
- **Chrome:** This newer browser was created by Google. It is highly standards-compliant, and it's especially powerful at handling Java Script and AJAX. This is not surprising, considering Google is one of the companies that pioneered the use of AJAX and is actively promoting its use. Chrome has one of the fastest JavaScript interpreters in common use.
- **Other browsers:** There are many other browsers in use today, including specialty browsers on various forms of Linux, cell phones, and PDAs. It is nearly impossible to support them all, but many browsers now at least try to support Web standards.

TIP

I prefer to do most of my testing with Firefox 3, because it has very good standards support and an excellent set of tools for improving and debugging your code. I then check my pages on other browsers including IE6, IE7, and Chrome.

Turning Firefox into a Development Machine

Firefox is an especially important browser for Web developers. It has a number of attractive features including its excellent standards support in HTML and JavaScript. However, the most important advantage of Firefox as a developer's tool might be its support for extensions. Many commercial browsers keep their code a closely guarded secret, and are very difficult to extend. Firefox was designed from the beginning to have its capabilities extended — and a number of very clever programmers have added incredible extensions to the tool. A few of these extensions have become "must haves" for Web developers:

Web Developer Toolbar

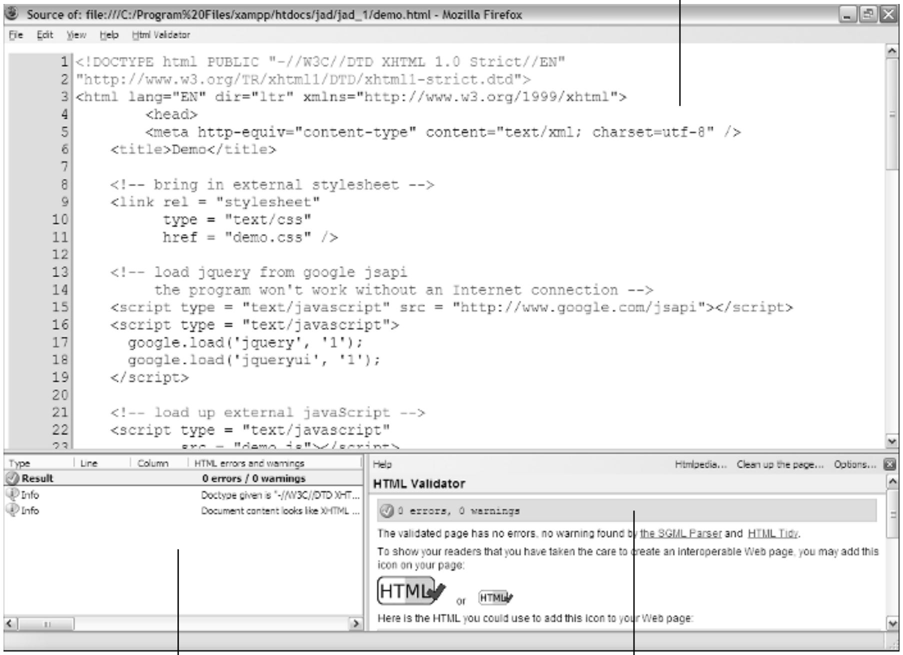
The Web Developer Toolbar by Chris Pederick is an incredible tool. It adds a new toolbar to Firefox with a number of extremely useful capabilities:

- **Edit CSS:** You can pop up a small window and type in CSS code. The CSS will take effect immediately in real time, so you can see exactly what effect your CSS has on the view of the page.
- **Display Ruler:** This incredibly handy tool allows you to draw a ruler on your page to see exactly how large various elements are. This is really useful for debugging Web layouts.
- **Outline tables:** This tool helps you make sense of table-based layouts. It's a good way to see how a complex table-based design is created.

  It's best to avoid table-based layout, but sometimes you have to look at somebody else's pages.
- **Resize menu:** The Resize menu lets you see how your page looks in a number of other standard sizes. This can be very useful when you're designing a layout.
- **Validation tools:** Web Developer includes a number of really handy tools for validating your Web pages. It includes links for validating HTML and CSS, as well as the primary accessibility standards.

HTML Validator extension

This incredible extension brings the same validation engine used by the W3C to your browser. It gives quick feedback on the validity of any page you view. It also adds much more information to Firefox's View Source page, including feedback on exactly which validation errors you have. (Validation information is not provided by the normal View Source page.) The hints for fixing the errors are actually helpful, and there's also a tool for automatically repairing the code with the excellent HTML Tidy program. HTML is the foundation for your JavaScript code, and invalid HTML provides a faulty framework. With the HTML Validator, there's no reason to have invalid pages. Figure 1-12 shows the improved View Source window with the warnings tab and the buttons for fixing the code with HTML Tidy.


View Source window
Source of: file:///C:/Program%20Files/xampp/htdocs/jad/jad_1/demo.html - Mozilla Firefox
File Edit View Help Html Validator
Type Line Column HTML errors and warnings
Help Htmlpedia... Clean up the page... Options...
HTML Validator
HTML or HTML
Validator errors appear in this window.
The Validor extension provides helpful error messages.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html lang="EN" dir="ltr" xmlns="http://www.w3.org/1999/xhtml">
        <head>
        <meta http-equiv="content-type" content="text/xml; charset=utf-8" />
    <title>Demo</title>

    <!-- bring in external stylesheet -->
    <link rel = "stylesheet"
          type = "text/css"
          href = "demo.css" />

    <!-- load jquery from google jsapi
         the program won't work without an Internet connection -->
    <script type = "text/javascript" src = "http://www.google.com/jsapi"></script>
    <script type = "text/javascript">
      google.load('jquery', '1');
      google.load('jqueryui', '1');
    </script>

    <!-- load up external javaScript -->
    <script type = "text/javascript"
            src = "demo.js"></script>
```

Type	Line	Column	HTML errors and warnings
Result			0 errors / 0 warnings
Info			Doctype given is "-//W3C//DTD XHT...
Info			Document content looks like XHTML ...

0 errors, 0 warnings

The validated page has no errors, no warning found by the SGML Parser and HTML Tidy.

To show your readers that you have taken the care to create an interoperable Web page, you may add this icon on your page:

Here is the HTML you could use to add this icon to your Web page:

Figure 1-12: The HTML Validator extension adds very useful features to the View Source tool.

Firebug

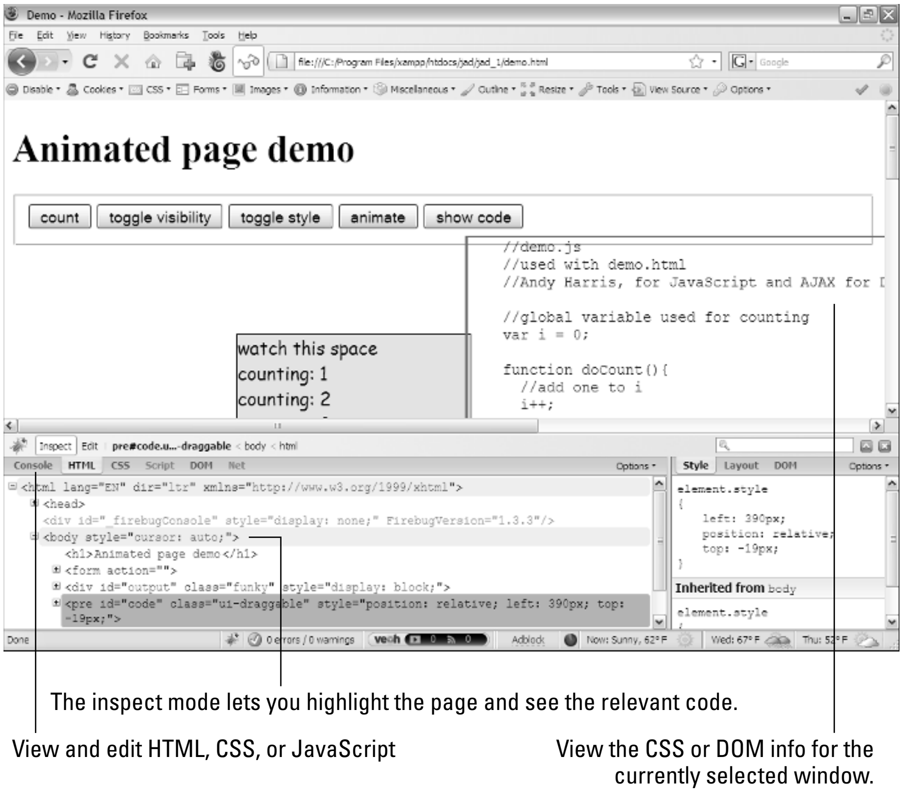
The Firebug extension is one of the most important tools in Web development. It turns Firefox into a complete debugging tool. Firebug has several especially useful features:

- **Inspect Window:** This incredible tool allows you to move your mouse over any element in your page and instantly see the code that created that section in another panel. This is a very easy way to analyze and debug pages. You can also see instantly what CSS applies to a particular snippet of code, and highlight code to see the corresponding output.
- **CSS View and Edit:** You can look over the CSS of your page in a panel, see previews of any colors, and edit the values. You'll see the results in real time on the page.
- **JavaScript Debugging:** Even pros make mistakes — and up to now, few debugger tools have been available for JavaScript programmers. Firebug has better mechanisms for error-trapping than the browsers do, and it

also incorporates a very nice debugger that can really help you find your errors as your program runs.

- **AJAX monitoring:** AJAX programming is based on a series of requests back and forth from the server. Firebug helps you to keep track of these requests and watch your data move.
- **Live code view:** The ordinary view source menu of most browsers helps you see the code as it originally comes from the browser. In JavaScript programming, you're often changing the page code on the fly. Firebug shows you the page as it really is, even if it's being changed by JavaScript. This is a very useful facility.
- **Firebug lite:** This is a variation of firebug that works in IE and other browsers. This adds most of the power of firebug to any browser.

Figure 1-13 shows Firebug in inspect mode. As the user moves over a piece of the page, the related code segment appears in the code window.


Demo - Mozilla Firefox
File Edit View History Bookmarks Tools Help
file:///C:/Program Files/xampp/htdocs/jad/jad_1/demo.html
Animated page demo
count
toggle visibility
toggle style
animate
show code
watch this space
counting: 1
counting: 2
Inspect Edit pre#code.u...-draggable < body < html
Console HTML CSS Script DOM Net Options
Style Layout DOM Options
Inherited from body
Done 0 errors / 0 warnings
The inspect mode lets you highlight the page and see the relevant code.
View and edit HTML, CSS, or JavaScript
View the CSS or DOM info for the currently selected window.

```
//demo.js
//used with demo.html
//Andy Harris, for JavaScript and AJAX for D

//global variable used for counting
var i = 0;

function doCount(){
  //add one to i
  i++;
```

```
<html lang="EN" dir="ltr" xmlns="http://www.w3.org/1999/xhtml">
  <head>
  <div id="_firebugConsole" style="display: none;" FirebugVersion="1.3.3"/>
  <body style="cursor: auto;">
    <h1>Animated page demo</h1>
    <form action="">
    <div id="output" class="funky" style="display: block;">
    <pre id="code" class="ui-draggable" style="position: relative; left: 390px; top:
    -19px;">
```

```
element.style
{
    left: 390px;
    position: relative;
    top: -19px;
}
Inherited from body
element.style
```

Figure 1-13: Firebug being used to inspect a Web page.