1

Introducing Cascading Style Sheets

Cascading style sheets is a language intended to simplify website design and development. Put simply, CSS handles the *look and feel* of a web page. With CSS, you can control the color of text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, as well as a variety of other visual effects.

CSS was created in language that is easy to learn and understand, but it provides powerful control over the presentation of a document. Most commonly, CSS is combined with the markup languages HTML or XHTML. These markup languages contain the actual text you see in a web page — the hyperlinks, paragraphs, headings, lists, and tables — and are the glue of a web document. They contain the web page's data, as well as the CSS document that contains information about what the web page should look like, and JavaScript, which is another language that provides dynamic and interactive functionality.

HTML and XHTML are very similar languages. In fact, for the majority of documents today, they are pretty much identical, although XHTML has some strict requirements about the type of syntax used. I discuss the differences between these two languages in detail in Chapter 2, and I also provide a few simple examples of what each language looks like and how CSS comes together with the language to create a web page. In this chapter, however, I discuss the following:

- The W3C, an organization that plans and makes recommendations for how the web should function and evolve
- How Internet documents work, where they come from, and how the browser displays them
- An abridged history of the Internet
- Why CSS was a desperately needed solution
- The advantages of using CSS

The next section takes a look at the independent organization that makes recommendations about how CSS, as well as a variety of other web-specific languages, should be used and implemented.

Who Creates and Maintains CSS?

Creating the underlying theory and planning how cascading style sheets should function and work in a browser are tasks of an independent organization called the World Wide Web Consortium, or W3C. The W3C is a group that makes recommendations about how the Internet works and how it *should* evolve. I emphasize *should*, because the World Wide Web Consortium has no control over the implementation of the standards that it defines. The W3C is comprised of member companies and organizations that come together to create agreed-upon standards for how the web should function. Many prominent companies and organizations are W3C members, including Microsoft, Adobe, The Mozilla Foundation, Apple, Opera Software, and IBM. The W3C oversees the planning of several web languages including CSS, HTML, XHTML, and XML, all of which are mentioned in this book.

CSS is maintained through a group of people within the W3C called the CSS Working Group. The CSS Working Group creates documents called *specifications*. When a specification has been discussed and officially ratified by W3C members, it becomes a recommendation. These ratified specifications are called *recommendations* because the W3C has no control over the actual implementation of the language. Independent companies and organizations create that software.

The specifications created by the W3C are not limited only to web browsers; in fact, the specifications can be used in a variety of software, including word processor and spreadsheet applications, as well as by different types of hardware devices, such as PDAs and cell phones. For that reason, the software implementing a specification is referred to by the W3C as the *user agent*, which is a generic term that encompasses all the different types of software that implement W3C specifications.

The W3C merely recommends that a language be implemented in a certain way to ensure that the language does what is intended no matter which operating system, browser, or other type of software is being used. The goal of this standardization is to enable someone using the Netscape browser, for example, to have the same Internet experience as someone using Internet Explorer, and likewise, for developers to have a common set of tools to accomplish the task of data presentation. Were it not for web standards, developing documents for the web might require an entirely different document for a given user agent. For example, Internet Explorer would require its own proprietary document format, while Mozilla Firefox would require another. Common community standards provide website developers with the tools they need to reach an audience, regardless of the platform the audience is using.

As I write this, CSS comes in four different versions, each newer version building on the work of the last. The first version is called CSS level 1, and became a W3C recommendation in 1996. The second version, CSS level 2, became a W3C recommendation in 1998. The third version, CSS level 2.1, is currently a working draft, downgraded from a candidate recommendation since I wrote the first edition of this book in 2004. A *candidate recommendation* is the status the W3C applies to a specification when it feels the specification is complete and ready to be implemented and tested. After the specification has been implemented and tested by at least a few of the member companies, the candidate recommendation is then more likely to become a full recommendation. A *working draft* is the status the W3C applies to an ongoing work, which is subject to change. The fourth version of CSS is called CSS level 3, and many portions of it are still in development. Although portions of CSS are officially subject to change by the W3C

standards body, I may discuss these features anyway if at least one browser maker has implemented the feature in question. I preface any such discussion with the warning that these features are still under development and could be subject to change.

This book discusses the portions of CSS available in browsers at the time of this writing — that includes most of CSS 2 and CSS 2.1, and a little of CSS 3. Some portions of CSS 2.1 contradict CSS 2 and are not yet implemented in any browser. Where appropriate throughout the book and before introducing a new CSS feature, I reference the W3C specification in which that CSS feature is documented by including the phrase *Documented in CSS* followed by the version number. Later in this chapter, I discuss the browsers that you need to test and build CSS-enabled web documents.

You can find the W3C website at `www.w3.org`. Go there to find documents that browser makers refer to when they are looking to implement languages such as CSS into a browser or other software. Be advised, these specifications lean heavily toward the technical side. They aren't intended as documentation for people who use CSS; rather, they are aimed at those who write programs that interpret CSS. Despite the heavily technical nature of the W3C specification documents, many web developers refer to the W3C documents as end-user documentation anyway, since it is the most complete resource.

Now that you know a little about who is responsible for planning and outlining the development of CSS, the next section describes how a web document makes its way into your browser.

How the Internet Works

As you probably already know, the Internet is a complex network of computers. Most of what goes on behind the scenes is of little interest to the person developing content for a website, but it is important to understand some of the fundamentals of what happens when you type an Internet address into your browser. Figure 1-1 shows a simple diagram of this process.

At the top of the diagram in Figure 1-1, you see a computer labeled *server-side* and a computer labeled *client-side.* The diagram is by no means an exhaustive or complete picture of what happens when you type in an Internet address, but it serves the purpose of illustrating the portions of the process that the aspiring web designer needs to understand. The server-side computer houses the documents and data of the website and is generally always running so that the website's visitors can access the website at any time of day. The client-side computer is, of course, your own computer.

The server-side computer contains HTTP server software that handles all the incoming requests for web pages. When you type an Internet address into a browser, the browser sends out a request that travels through a long network of computers that act as relays for that request until the address of the remote (server-side) computer is found. After the request reaches the HTTP server, the HTTP server sees what it is you are trying to find, searches for the page on the server's hard drive, and responds to the request you've made, sending the web page that you expect. That response travels back through another long chain of computers until your computer is found. Your browser then opens the response and reads what the HTTP server has sent back to it. If that server has sent an HTML document or another type of document that your browser can interpret, it reads the source code of that document and processes it into a displayable web page.

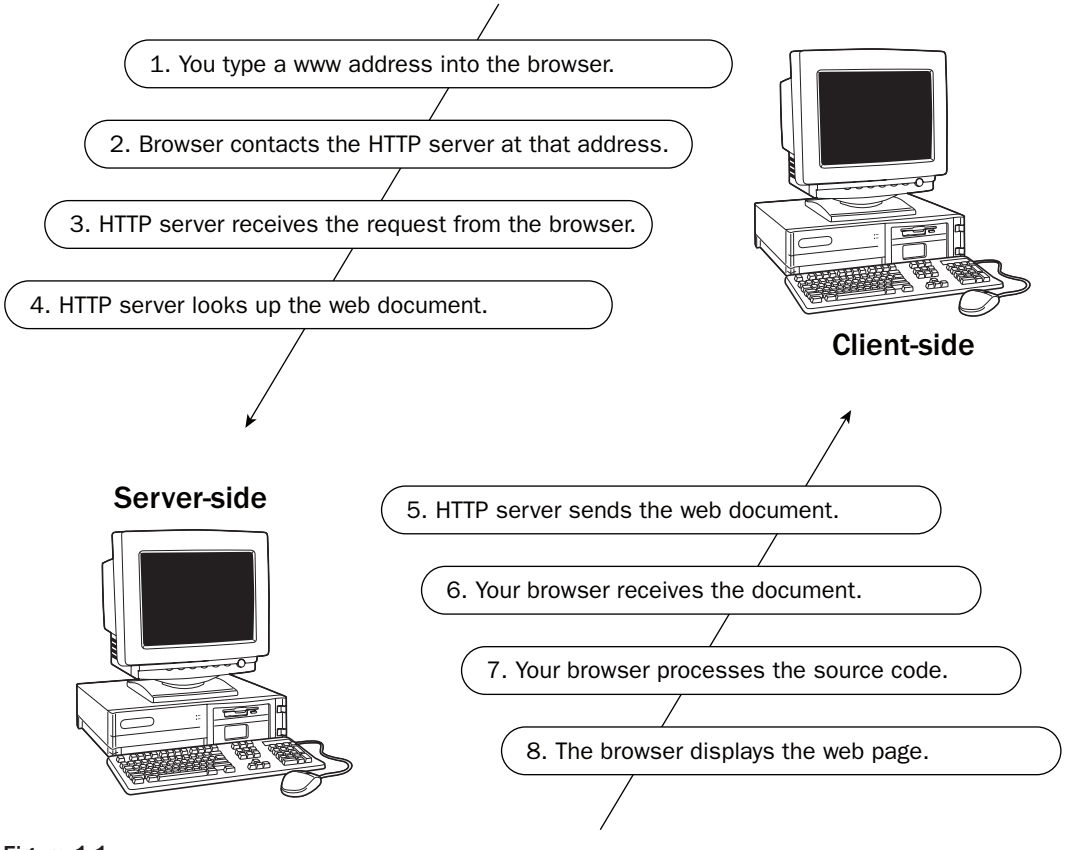

Figure 1-1

This is where CSS enters the picture. If CSS is present in the document, the CSS describes what the HTML page should look like to the browser. If the browser understands the CSS, it processes the web page into something you can see and interact with. If the browser understands only some of the CSS, it generally ignores what it doesn't understand. If the browser doesn't understand CSS at all, it usually displays a plain-looking version of the HTML document.

How CSS Came to Be

During the mid-1990s, use of the Internet exploded. At that time, HTML was the only option for presenting a web page. As the Internet began to be used by more regular folks (as opposed to government, educational institutions, and researchers, as in the early days), users began demanding more control over the presentation of HTML documents. A great quandary arose — clearly HTML alone was not good enough to make a document presentable. In fact, not only was it not good enough, HTML alone simply wasn't suited for the job. HTML did not have the functionality that professional publishing required and had no way of making magazine- or newspaper-like presentations of an electronic document.

At the time, style sheets were not a new invention. In fact, style sheets were part of the plan from the beginning of HTML in 1990. Unfortunately, however, no standardized method of implementing style sheets was ever outlined, leaving this function up to the various browsers. In 1994, Tim Berners-Lee founded the World Wide Web Consortium, and a few days later, Håkon Wium Lie published his first draft of Cascading HTML Style Sheets. This draft was a proposal for how HTML documents could be styled using simple declarations.

Of those that responded to Håkon's draft of Cascading HTML Style Sheets was Bert Bos, who was working on a style sheet proposal of his own. The two joined forces and came up with cascading style sheets. They dropped HTML from the title, realizing that CSS would be better as a general style sheet language, applicable to more than one type of document. CSS caused some controversy at its inception because part of the underlying fundamentals of the new style sheet language was that it created a balance between the browser's style sheet, the user's style sheet, and the author's style sheet. Some simply didn't like the idea that the user could have control over the presentation of a web document. Ultimately, however, the Internet community accepted CSS.

Among CSS supporters was Microsoft, who pledged support for the new style sheet language in its Internet Explorer web browser. Netscape, on the other hand, another popular web browser at the time, remained skeptical about CSS and went forward with a style sheet language of its own called JavaScript Style Sheets, or JSSS. Ultimately, Netscape's style sheets were not successful. Eventually, because of a series of bad decisions and setbacks on the part of Netscape as a whole and Netscape's management, Netscape ultimately began losing market share, and Microsoft's Internet Explorer (IE) browser grew more and more popular. At IE's peak, it held 95 to 98 percent of the browser market share. Although IE has since lost market share to the likes of Mozilla Firefox and Safari, at the time of this writing, IE is still the dominant browser, most firms putting IE's market share at 50 to 85 percent, depending on the website's audience. Mainstream sites will see upward of 85 percent, but technical websites may see around 50 percent. Your own website's browser statistics will depend largely on the content of your site. One such site to reference for statistics is `http://www.upsdell.com/BrowserNews/stat.htm`. However, keep in mind the quote, "There are lies, damn lies — and statistics" — Disraeli (later made famous by Mark Twain).

During the time that CSS was being planned, browsers began allowing HTML features that control presentation of a document into the browser. This change is the primary reason for much of the bloated and chaotic source code in the majority of websites operating today on the Internet. Even though HTML was never supposed to be a presentational language, it grew to become one. Unfortunately, by the time CSS level 1 was made a full W3C recommendation in 1996, the seed had already been planted. Presentational HTML had already taken root in mainstream website design and continues today.

However, all is not lost. Today, the most popular browsers have fantastic support for cascading style sheets. Ironically, the browser exhibiting the least support is Microsoft's Internet Explorer for Windows, which still has plenty of CSS support to do away with most presentational HTML design. More ironic still, among the browsers with the best CSS support is Netscape's browser, and its open source offspring, Mozilla Firefox. This may beg the question: If Microsoft was such an avid supporter of cascading style sheets in the beginning, why is Microsoft's browser the least standards-compliant today? The answer is that Microsoft did indeed follow through with its promise for CSS support, and it was the most comprehensive and impressive implementation of CSS even up to the release of Internet Explorer 6 in 2001. Even so, CSS implementation in Internet Explorer has declined since the release of Internet Explorer 5. We can only speculate as to why Microsoft's browser declined in its support for CSS.

In the next section, I talk about the different types of browsers that you'll need to work through the examples for this book.

Browsers

Because CSS is a standard web language, many browsers support it. Therefore, it stands to reason that the aspiring web designer would want to harness that standardization to reach the largest audience possible, regardless of operating system or platform. In this section I provide an overview of each of these browsers, and where you can look to obtain a new version of that browser. Together, the following browsers combined comprise over 99 percent of the browser market share for the majority of websites in operation today:

- Internet Explorer 6 and 7 for Windows
- Mozilla Firefox for Windows, Mac, and Linux
- Opera for Windows, Mac, and Linux
- Safari for Mac OS X

In the next section, I discuss Internet Explorer 6 and 7 for Windows.

Internet Explorer

Internet Explorer is Microsoft's flagship browser that comes preloaded with the Windows operating system. The current stable version, as of this writing, is version 7.

Internet Explorer 7

Late in 2004, after the first edition of this book was published, Microsoft finally began work on a new version of Internet Explorer. IE 7 includes stronger security, tabbed browsing, and other goodies for users, and for developers — improvements to IE's support for CSS!

IE 7 comes just over five years after the release of IE 6, which was released in 2001. IE 7 is a fantastic improvement over IE 6, but it still doesn't quite meet the level of CSS present in competing browsers like Apple's Safari browser, or Mozilla Firefox. Although it doesn't exhibit the best CSS support, there is hope that future versions of IE will make significant progress in this area. Internet Explorer developers, and even Bill Gates, have publicly stated that Microsoft has returned to a more frequent release cycle for Internet Explorer, and we can expect new versions of Internet Explorer every year for the foreseeable future. Microsoft has even gone so far as to admit that it made a mistake by waiting too long to release a new version of IE.

Even though IE 7 is finally here, it will be years still before it achieves sufficient market penetration that web developers can officially dump support for IE 6. Because of IE 6's deficiencies, many are chomping at the bit for that time to come. In the meantime, we'll have to continue to support it and work around its shortcomings.

IE 7 is available for the following operating systems:

- Windows XP Service Pack 2
- Windows XP Pro 64-bit Edition
- Windows Server 2003
- Windows Vista

You can obtain IE 7 from Microsoft's website at `http://www.microsoft.com/ie`.

Internet Explorer 6

At the time of this writing Internet Explorer 6 is still the top dog with its browser market share between 50 and 85 percent, depending on the website's audience (see my discussion of Internet Explorer's market share in the section titled "How CSS Came to Be" earlier in this chapter). If you don't already have IE 6, you can obtain it from `http://www.microsoft.com/windows/ie/ie6/default.mspx`.

Installing Multiple Versions of Internet Explorer for Testing

At the time of this writing, you cannot install IE 7 alongside IE 6 on the same copy of Windows. For development, you need a way to test IE 6 and IE 7 both, since you'll have visitors to your website on both browsers. The following are a few ways to do this.

- Use PC virtualization/emulation software such as Virtual PC (a product made by Microsoft), which allows you to install and run different versions of Windows (or other operating systems, such as Linux) from within Windows or Mac OS X. Essentially, you can load up a new instance of Windows from your Windows desktop, and have that instance of Windows run in a window, independently. For example, Figure 1-2 shows a screenshot of me running Windows XP and IE 6 from my Mac OS X desktop, using the open source software Q, which lets me install and run Windows from within Mac OS X.
- Another option is setting up two different physical computers, one with IE 6 installed, and the other with IE 7.
- If you're feeling particularly adventurous, you can set up two installations of Windows on the same computer, although for this discussion, this method is a bit too advanced for me to adequately cover here. If you'd like to learn more about installing Windows more than once on the same computer, more information about that can be found at `http://www.microsoft.com/windowsxp/using/setup/learnmore/multiboot.mspx`.

Figure 1-2 shows two instances of Windows XP running in Parallels Desktop for Mac; one is running IE 6, and the other is running IE 7.

Most people prefer to keep it simple, and have all of their development tools at their fingertips. That makes the virtualization/emulation method the most attractive, in lieu of actually being able to install IE 6 alongside IE 7. I discuss this method in further detail in the next section.

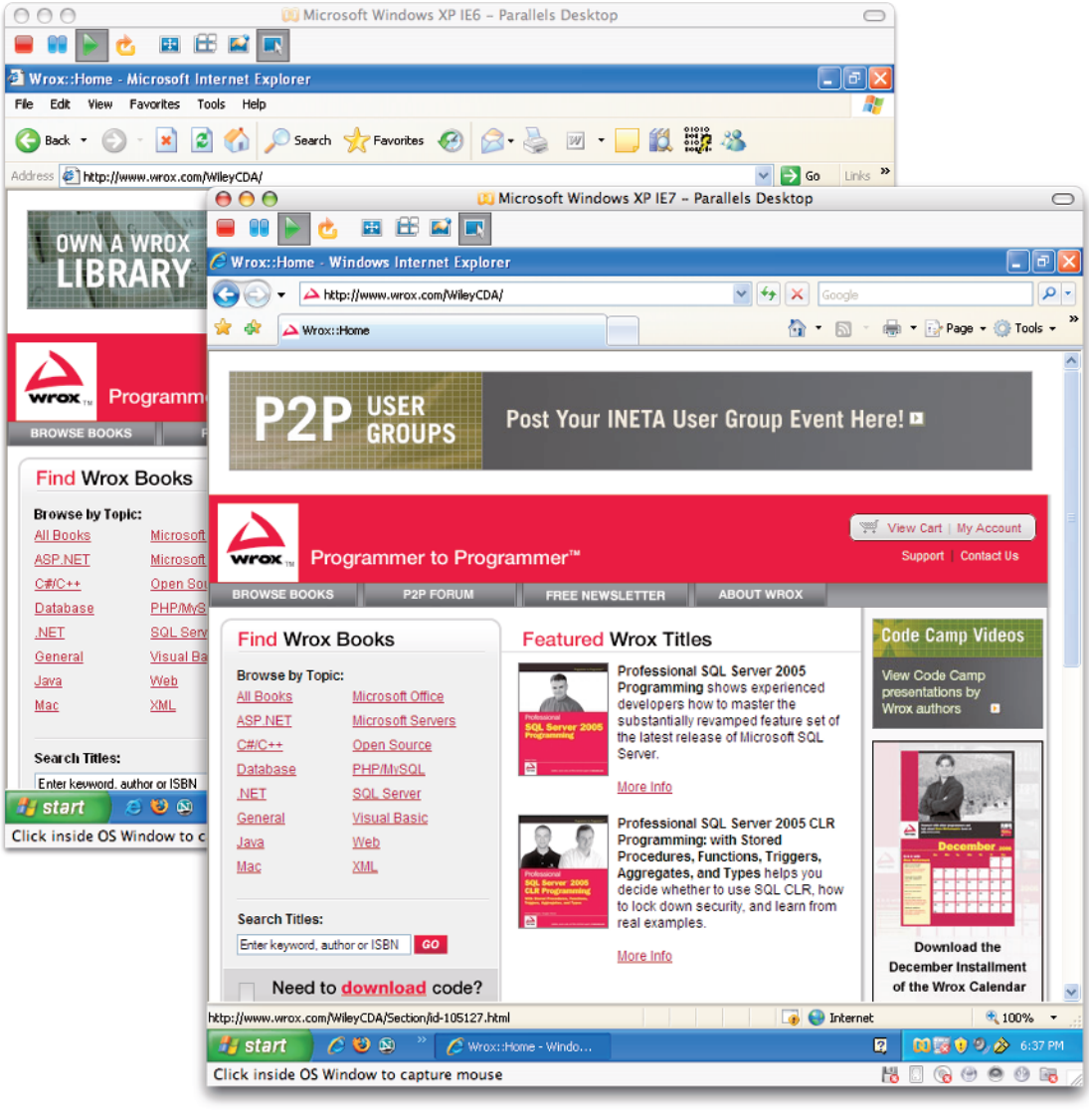

Figure 1-2

Installing Windows Using PC Virtualization/Emulation Software

Today many companies make PC virtualization or emulation software, which allows you to run an entire operating system from a window on your desktop in the manner illustrated in Figure 1-2. More or less, it's like having multiple computers all consolidated into one. You can boot up a virtual computer, with all default settings so you can test your web pages. Here are some of the titles available.

- **VMWare, Player:** Available for free from `http://www.vmware.com/products/player/` for Windows and Linux.
- **Virtual PC:** Made by Microsoft, available for $129 from `http://www.microsoft.com/windows/virtualpc` (the price does not include a license for running Windows in the Virtual PC). Requires Windows or a PowerPC-based Mac.
- **Q** (pictured in Figure 1-2)**:** Available for free from `http://www.kberg.ch/q`. If you're using Mac OS X, Q is available as a universal application (it runs on both PowerPC-based and Intel-based Macs).
- **Parallels:** Available for $49.99 from `http://www.parallels.com` for Windows, Mac (PowerPC and Intel-based), and Linux.

The best software for installing Windows from another operating system is software that uses *virtualization*. Without going into the technical details, software using virtualization runs much faster. The other, slower, much slower, in fact, method is *emulation*. Parallels and VMWare use virtualization, whereas, at the time of this writing, Microsoft's Virtual PC and "Q" both use emulation. Your computer will also need serious horsepower to run two operating systems at the same time; see each respective website for the system requirements of each of the aforementioned solutions. In my experience, software like this works best with at least 1GB of RAM and about a 2 GHz processor.

Without the ability to install and work with Windows virtually using software such as VMWare, your last resort is to uninstall IE 7 every time you need to test in IE 6, which can throw a pretty big wrench in the testing process. Currently, the virtual machine solution is the one officially sanctioned and recommended by Microsoft for testing in multiple versions of Internet Explorer. The IE team has responded to requests from web developers for the ability to install and run multiple versions of Internet Explorer side-by-side, and have said they are looking at the problem, but have not yet publicly announced a solution or released software to remedy the problem.

Internet Explorer for PowerPC Mac OS X

For PowerPC Macintosh users, I recommend not using or testing in IE for Mac. The capabilities and bugs of IE for Windows and IE for Mac are very different. IE for the Macintosh has better support for CSS (compared to IE 6), but it is an entirely different browser. The name may be the same, but the browsers are very different. In fact, Microsoft has completely dropped support for IE for Mac, having stopped development with a public announcement made in 2003, and having completely stopped support in 2005. It has less than a tenth of a percent of market share, if that much, and it does not run on Apple's Intel-based Macs.

For Mac users, I recommend Apple's own Safari or a Gecko browser, such as Camino or Mozilla Firefox, which I discuss further in the coming sections. If you don't have Internet Explorer for Windows, you still can work through most exercises and examples presented in this book, but if you are serious about website design, you will need to find a way to test your websites in Internet Explorer on Windows, since it has the majority of market share, and will enjoy that status far into the foreseeable future.

For news on what is transpiring in the world of Internet Explorer development, you might like to check out the Internet Explorer Team's blog at `http://blogs.msdn.com/ie`. New IE features and news of anything relating to Internet Explorer are announced on the IE Team blog.

The Gecko Browsers: Mozilla Firefox, Netscape, Camino

Gecko was created in January 1998. At that time, Netscape announced that it was making its browser free to its users and that its browser would be *open source*, meaning that its source code would be freely available for modification and distribution. This led to the creation of Mozilla; at the time Mozilla was the organization charged with developing and managing the Netscape code base. America Online later purchased Netscape, and until July 2003 Mozilla remained a part of Netscape. In July 2003, the Mozilla Foundation was created, making Mozilla an independent, not-for-profit corporation. When the Netscape browser became open source, its rendering engine, the part of the browser software responsible for making the source code of a web page into something you can see and interact with, was given the name Gecko.

Gecko is the foundation that a whole suite of browsers relies on to do the behind-the-scenes work of rendering web pages. Gecko is included in AOL for Mac OS X, Camino, Netscape 6, Netscape 7, Netscape 8, Mozilla Suite, Mozilla Sea Monkey, and Mozilla Firefox.

Netscape's browser market share has greatly diminished, whereas Mozilla Firefox continues to gain in popularity, occupying the number-two spot at between 5 and 30% market share (again, depending on the website's audience). Netscape's (and other Gecko browsers, for that matter) market share is charted by most statistics at less than one percent.

The following table shows the relationship between other Gecko browsers and Mozilla Firefox. This table illustrates the version of the underlying Gecko engine that each browser has in common with Firefox. Each of these browsers can be expected to render a web page identically and have the same capabilities in the area of CSS and document layout as the version of Firefox cited.

Other Gecko Browser	Firefox
Netscape 8.1	Firefox 1.5
Netscape 8.0	Firefox 1.0
Netscape 7.2	Firefox 0.9
Camino 1.0	Firefox 1.5
SeaMonkey 1.0 (formerly Mozilla Suite)	Firefox 1.5
Mozilla Suite 1.8	Firefox 1.0
Mozilla Suite 1.7	Firefox 0.9
Mozilla Suite 1.6	Firefox 0.8

Netscape 8.0 and 8.1 both feature the ability to switch between IE and Gecko for rendering a web page from within the Netscape browser, so essentially it is both Internet Explorer and Gecko in the same browser. The version of Internet Explorer in Netscape 8.0 and 8.1 is the same as the version of IE installed on the system. Netscape uses Gecko by default, but may try to "automatically" select the best rendering engine to use for a given website.

You can see that Firefox 0.9 and Mozilla Suite 1.7 can be expected to behave identically where CSS and design layout is concerned.

> *Because gecko browsers share the same brain (and because of Firefox's popularity), for the remainder of this book, I cite only Firefox when referring to a Gecko browser.*

Depending on which Gecko browser you happen to like, you can obtain Gecko browsers from the following places:

- **Mozilla Firefox for Windows, Mac, and Linux:** Available from `http://www.mozilla.com/firefox`
- **Netscape for Windows:** Available from `http://www.netscape.com/download`
- **Camino for Mac:** Available from `http://www.caminobrowser.org/`
- **SeaMonkey for Windows, Mac, and Linux:** Available from `http://www.mozilla.org/projects/seamonkey/`

Safari

The next browser that I discuss is Safari, which is based on Konqueror, an open source browser available for Linux operating systems. The rendering engine used in the Safari and Konqueror web browsers is called KHTML. While Konqueror and Safari both have KHTML in common, Safari is a fork of KHTML (a *fork* means they shared the exact same source code at one point, but now each is developed independently), and features found in Safari may not necessarily appear in Konqueror and vice versa. Despite this, the two browsers render documents very similar to one another. Apple develops Safari, independently of Konqueror, and is the browser included with Macintosh OS X operating systems. Before Safari, Internet Explorer for Mac and Gecko had been dominant on the Mac.

For the purpose of this book, I note Safari compatibility when appropriate. Safari is available only for Mac OS X and can be obtained from `www.apple.com/safari`. Konqueror is only available for Linux (and any operating system in which KDE, the K Desktop Environment, runs) at the time of this writing; it can be found at `www.konqueror.org`.

Opera

Opera is a lesser-known, Norwegian-based company. Opera users are fewer, accounting for only a few percent market share by most statistical estimates. Again, that figure can be much higher or lower depending on a website's audience. Also be aware that Opera and Mozilla Firefox browsers can be configured to identify themselves to a website as Microsoft Internet Explorer browsers. This, of course, can distort statistical analysis. This spoofing is done because websites often create content targeting Microsoft Internet Explorer and Netscape specifically, leaving everyone else out in the cold — even though third-party browsers like Mozilla Firefox and Opera probably support the required functionality.

At the time of this writing, the current version of the Opera browser is 9.0. You can download this browser for free from `www.opera.com`. The Opera browser is available for Windows, Macintosh, Linux, and a variety of other platforms.

Writing CSS

To write CSS, just as is the case when writing HTML source, you will need a text editor. WYSIWYG (What You See Is What You Get) editors such as Microsoft Word aren't ideally suited for CSS because the environment is not ideal for the composition of source code. WYSIWYG programs often have features like AutoCorrection and line wrapping; a plain text editor is more appealing precisely because it does not have these automatic features. Furthermore, the more automated WYSIWYG editors are designed to write the source code for you behind the scenes, so you don't have complete control over the structure and formatting of the source code. In contrast, a plain text editor doesn't insert anything into the source code beyond what you type into the text editor.

The Windows Notepad program is one example of a text editor that is ideal for composing source code. To launch Notepad, choose Start ➪ Run and then type **Notepad** in the Open text box. You can also use Microsoft FrontPage, but FrontPage is best used in *view source* mode where you can edit the source code directly instead of via the WYSIWYG interface. The same holds true for Macromedia Dreamweaver.

On Mac OS X, the Notepad equivalent is TextEdit, which can be found in the Mac OS X Applications folder.

If Notepad or TextEdit is just too basic for your taste, a text editor that highlights markup and CSS syntax might suit your needs better. The following are full-featured alternative text editors for Windows:

- Crimson Editor: `www.crimsoneditor.com` (free)
- HTML-kit: `www.chami.com/html-kit` (free)

Here are some alternative text editors that work with Mac OS X:

- CreaText: `http://creatext.sourceforge.net` (free)
- BBEdit: `www.barebones.com` (shareware)

If you're using Linux, you're probably already familiar with the different text editors that come bundled with the various distributions.

You must create HTML files with the `.html` extension. If you use Notepad or TextEdit, beware of your files being saved with a `.txt` extension, which will not result in the HTML file you were going for.

To ensure that your files are saved properly on Windows, choose Start ➪ Run and type **Explorer** (or right-click Start and choose Explore from the pop-up menu) to open Windows Explorer. After Windows Explorer is open, choose Tools ➪ Folder Options to open the Folder Options window, click the View tab, and uncheck the Hide Extensions for Known File Types box (see Figure 1-3). Then click OK.

> *HTML files are not the only file type in which the document extension is important; other file types require specific extensions as well. Those file types are covered later in this chapter.*

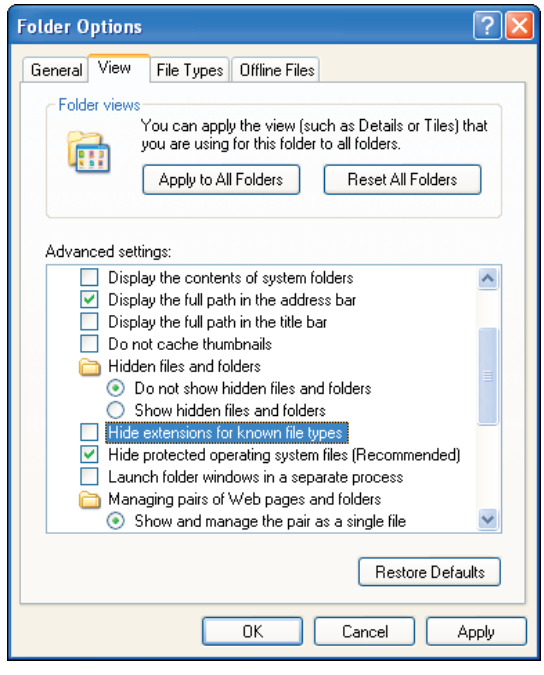

Figure 1-3

On Mac OS X, open Finder, and go to Finder ➪ Preferences. Select the Advanced tab, and check the box for Show All File Extensions, which is depicted in Figure 1-4.

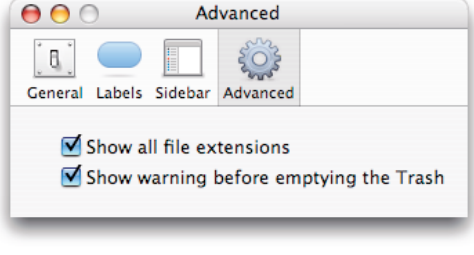

Figure 1-4

Armed with a browser and a text editor, in the next section I present an example of what CSS can do.

Your First CSS-Enabled Document

The following example is designed to introduce you to what CSS is capable of. It is designed to help you get your feet wet and get straight down to the business of writing style sheets.

You can find the images and source code for the following example at `www.wrox.com`*. While you can obtain the source code from* `www.wrox.com`*, I recommend that you type out the example so that you can get used to writing the syntax, and take in the different bits that come together in the example.*

Try It Out — Creating a CSS-Enabled Document

Example 1-1. To write your first CSS-enabled document, follow these steps.

1. In your text editor of choice, enter the following markup:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns='http://www.w3.org/1999/xhtml' xml:lang='en'>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>The Gas Giants</title>
    <link rel='stylesheet' type='text/css' href='solar_system.css' />
    <script type='text/javascript'>
      var fixpng = function($img) {};
    </script>
<!--[if lt IE 7]>
  <link rel='stylesheet' type='text/css' href='solar_system.ie.css' />
  <script type='text/javascript'>
  // This fixes PNG transparency in IE
  var fixpng = function($img)
  {
    var $html =
      '<span ' +
        (($img.id)?    "id='"  + $img.id + "' "    : '') +
        (($img.className)? "class='" + $img.className + "' " : '') +
        (($img.title)?   "title='" + $img.title + "' "   : '') +
        'style="' +
          'display: inline-block;' +
          'width: ' + $img.width + 'px;' +
          'height: ' + $img.height + 'px;' +
          "filter:progid:DXImageTransform.Microsoft.AlphaImageLoader(" +
          "src='" + $img.src + "', sizingMethod='scale'); " +
        $img.style.cssText + '" ';

    if ($img.getAttribute('mouseoversrc'))
    {
      $html += "mouseoversrc='" + $img.getAttribute('mouseoversrc') + "' ";
    }

    if ($img.getAttribute('mouseoutsrc'))
    {
      $html += "mouseoutsrc='" + $img.getAttribute('mouseoutsrc') + "' ";
    }

    $html += '></span>';

    $img.outerHTML = $html;
```

```
  }
  </script>
<![endif]-->
  </head>
  <body>
    <!--
       Image reuse guidelines:
       http://www.nasa.gov/multimedia/guidelines/index.html
    -->
<div id='solar-system'>
  <div class='planet jupiter'>
    <img src='images/jupiter.png'
       alt='Jupiter'
       class='planet'
       onload='fixpng(this);' />
    <div class='planet-copy'>
      <h1>Jupiter</h1>
      <ul>
        <li><b>Distance from the Sun:</b> 78,412,020 km</li>
        <li><b>Equatorial Radius:</b> 71,492 km</li>
        <li><b>Volume:</b> 1,425,500,000,000,000 km<sup>3</sup></li>
        <li><b>Mass:</b> 1,898,700,000,000,000,000,000,000,000 kg</li>
        <li>
<a href='http://solarsystem.jpl.nasa.gov/planets/profile.cfm?Object=Jupiter'>
  More Facts
</a>
        </li>
      </ul>
      <img src='images/symbols/jupiter.png'
         alt='Mythological Symbol for Jupiter'
         onload='fixpng(this);' />
    </div>
  </div>
  <div class='planet saturn'>
    <img src='images/saturn.png'
       alt='Saturn'
       class='planet'
       onload='fixpng(this);'/>
    <div class='planet-copy'>
      <h1>Saturn</h1>
      <ul>
        <li><b>Distance from the Sun:</b> 1,426,725,400 km</li>
        <li><b>Equatorial Radius:</b> 60,268 km</li>
        <li><b>Volume:</b> 827,130,000,000,000 km<sup>3</sup></li>
        <li><b>Mass:</b> 568,510,000,000,000,000,000,000,000 kg</li>
        <li>
<a href='http://solarsystem.jpl.nasa.gov/planets/profile.cfm?Object=Saturn'>
  More Facts
</a>
        </li>
      </ul>
      <img src='images/symbols/saturn.png'
```

```
          alt='Mythological Symbol for Saturn'
          onload='fixpng(this);' />
      </div>
    </div>
    <div class='planet uranus'>
      <img src='images/uranus.png'
         alt='Uranus'
         class='planet'
         onload='fixpng(this);' />
      <div class='planet-copy'>
        <h1>Uranus</h1>
        <ul>
          <li><b>Distance from the Sun:</b> 2,870,972,200 km</li>
          <li><b>Equatorial Radius:</b> 25,559 km</li>
          <li><b>Volume:</b> 69,142,000,000,000 km<sup>3</sup></li>
          <li><b>Mass:</b> 86,849,000,000,000,000,000,000,000 kg</li>
          <li>
<a href='http://solarsystem.jpl.nasa.gov/planets/profile.cfm?Object=Uranus'>
  More Facts
</a>
          </li>
        </ul>
        <img src='images/symbols/uranus.png'
           alt='Mythological Symbol for Uranus'
           onload='fixpng(this);' />
      </div>
    </div>
    <div class='planet neptune'>
      <img src='images/neptune.png'
         alt='Neptune'
         class='planet'
         onload='fixpng(this);' />
      <div class='planet-copy'>
        <h1>Neptune</h1>
        <ul>
          <li><b>Distance from the Sun:</b> 4,498,252,900 km</li>
          <li><b>Equatorial Radius:</b> 24,764 km</li>
          <li><b>Volume:</b> 62,526,000,000,000 km<sup>3</sup></li>
          <li><b>Mass:</b> 102,440,000,000,000,000,000,000,000 kg</li>
          <li>
<a href='http://solarsystem.jpl.nasa.gov/planets/profile.cfm?Object=Neptune'>
  More Facts
</a>
          </li>
        </ul>
        <img src='images/symbols/neptune.png'
           alt='Mythological Symbol for Neptune'
           onload='fixpng(this);'/>
      </div>
    </div>
  </div>
    </body>
  </html>
```

2. Save the preceding file in a new folder of its own as `index.html`.

3. Create a new, blank document in your text editor, and enter the following CSS:

```
body {
  margin: 0;
  padding: 0;
  background: #000 url('images/backgrounds/star.png') no-repeat fixed;
  font: 12px sans-serif;
}
a {
  text-decoration: none;
  color: lightblue;
}
a:hover {
  color: yellow;
}
div#solar-system {
  position: relative;
  height: 575px;
  margin: 50px 0 0 0;
  border-top: 1px solid #000;
  border-bottom: 1px solid #000;
  background: #000 url('images/backgrounds/star_darker.png') no-repeat fixed;
  overflow: auto;
  white-space: nowrap;
}
div.planet {
  position: absolute;
  top: 0;
  left: 0;
  bottom: 25px;
}
div.jupiter img.planet {
  margin: 75px 0 0 40px;
}
div.saturn {
  left: 900px;
}
div.uranus {
  left: 1900px;
}
div.uranus img.planet {
  margin: 175px 0 0 100px;
}
div.neptune {
  left: 2750px;
}
div.neptune img.planet {
  margin: 175px 0 0 200px;
}
div.planet img {
  float: left;
  margin-top: 20px;
```

```
}
div.planet-copy {
  color: white;
  padding: 10px;
  margin-left: 520px;
  background: #000 url('images/backgrounds/star_darker_still.png') no-repeat
fixed;
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  border-left: 1px solid #000;
  border-right: 1px solid #000;
}
div.planet-copy h1 {
  border-bottom: 1px solid #000;
  margin: 0 -10px;
  padding: 0 10px;
}
div.planet-copy ul {
  list-style: none;
}
```

4. Save the preceding CSS in the same folder where you saved `index.html`, as `solar_system.css`.
5. Enter the following CSS in a new document in your text editor:

```
div.planet {
  height: expression(document.getElementById('solar-system').offsetHeight - 25);
}
div.planet-copy {
  height: expression(document.getElementById('solar-system').offsetHeight - 45);
}
```

6. Save the preceding document in the same folder as `index.html` and `solar_system.css`, as `solar_system.ie.css`. The preceding source code results in the image in Figure 1-5, when loaded into Safari on Mac OS X.

   To see how `index.html` looks in other browsers, you can load it up by going to the File menu of the browser you'd like to view it in, and then select Open or Open File, and then locate `index.html` on your hard disk.

How It Works

Example 1-1 is an introduction to a little of what CSS is capable of. This example is designed to get your hands dirty up front with CSS, as a preview of what you can expect throughout the rest of the book. With each new chapter, I introduce and explain each of the nuts and bolts that come together to make examples like the preceding one. In Figure 1-5, you can see that CSS can be used to specify background images, and other aesthetic aspects of an XHTML document. I continue to revisit and explain the CSS that resulted in Figure 1-5 throughout the book.

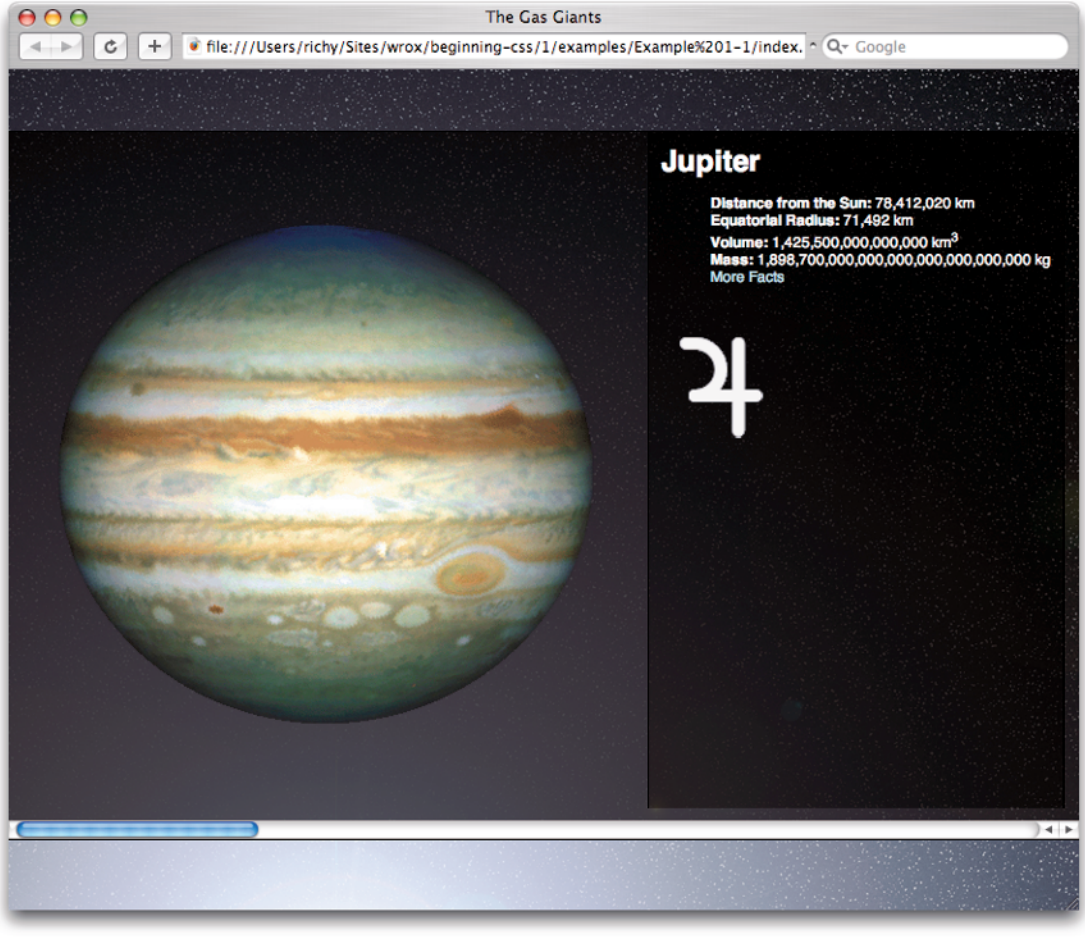

Figure 1-5

You might also note that Example 1-1 took some additional handy work to make it come out the same in Internet Explorer, as it did in Safari, Firefox, and Opera. Throughout this book, you also learn the hacks and workarounds that you need to make CSS-enabled web pages compatible with IE 6.

Advantages of Using CSS

By using cascading style sheets for the presentation of a web document, you can substantially reduce the amount of time and work spent on composing not only a single document, but an entire website. Because more can be done with less, cascading style sheets can reduce the amount of hard disk space that a website occupies, as well as the amount of bandwidth required to transmit that website from the server to the browser. Cascading style sheets have the following advantages:

- The presentation of an entire website can be centralized to one or a handful of documents, enabling the look and feel of a website to be updated at a moment's notice. In legacy HTML documents, the presentation is contained entirely in the body of each document. CSS brings a much needed feature to HTML: the separation of a document's structure from its presentation. CSS can be written independently of HTML.
- Users of a website can compose style sheets of their own, a feature that makes websites more accessible. For example, a user can compose a high-contrast style sheet that makes content easier to read. Many browsers provide controls for this feature for novice users, but it is CSS nonetheless.
- Browsers are beginning to support multiple style sheets, a feature that allows more than one design of a website to be presented at the same time. The user can simply select the look and feel that he or she likes most. This could only be done previously with the aid of more complex programming languages.
- Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
- Style sheets download much more quickly because web documents using CSS take up less hard disk space and consume less bandwidth. Browsers also use a feature called *caching*, a process by which your browser will download a CSS file or other web document only once, and not request that file from the web server again unless it's been updated, further providing your website with the potential for lightning-fast performance.

Cascading style sheets allow the planning, production, and maintenance of a website to be simpler than HTML alone ever could be. By using CSS to present your web documents, you curtail literally days of development time and planning.

Summary

Cascading style sheets are the very necessary solution to a cry for more control over the presentation of a document. In this chapter, you learned the following:

- The World Wide Web Consortium plans and discusses how the Internet should work and evolve. CSS is managed by a group of people within the W3C called the CSS Working Group. This group of people makes recommendations about how browsers should implement CSS itself.
- The Internet is a complex network of computers all linked together. When you request a web document, that request travels through that network to a computer called an HTTP server that runs software. It sends a response containing the page you requested back through the network. Your browser receives the response and turns it into something you can see and interact with.
- CSS answers a need for a style sheet language capable of controlling the presentation of not only HTML documents, but also several types of documents.
- Internet Explorer 6, Gecko, Opera, and KHTML browsers make up the majority of browsers in use today, with Internet Explorer 6 being the world's most popular browser.

- CSS has many advantages. These include being accessible, applicable to more than one language; applicable to more than one type of device, and allowing a website to be planned, produced, and maintained in much less time. CSS also enables a website to take up significantly less hard disk space and bandwidth than formerly possible.

Now that you have the tools to write CSS, and have seen a little of what CSS can do, in Chapter 2, I begin talking about the bits and pieces that come together in a CSS document to define the CSS language.