1

Getting Started

CSS (**C**ascading **S**tyle **S**heets) have been part of Web design for many years now, but it has been only within the past few years that the use of CSS has really exploded. More and more, Web designers are realizing the power and capabilities of CSS, and if you're reading this book, you most likely have at least a fundamental idea of what CSS can do for you and how it can make your life easier. This chapter discusses the purpose and effects of using CSS as well as which software applications you can use to create it.

Understanding Why You Need This Book

Before you start reading, it's probably fair to wonder what to expect. In this chapter, you'll learn all about this book.

Intended Audience

If you've been working with CSS for a while and are looking to boost your skills to expert status, this book isn't for you. But if you know how to write basic XHTML and you're just getting started with CSS, or have dabbled and would like to establish a solid grounding in CSS, this book is absolutely for you.

In general, the audience we envisioned while writing this book was a beginners' class—a "CSS 101" sort of crowd. If we were teachers at a night school or a local college and we needed our Web design class to learn CSS in order to make them better designers, this is the kind of book we'd want them to read. It doesn't spend pages and pages on every nuance of the technology. It isn't heavy on theory. It's a practical text, designed to instill a working knowledge of CSS, with just enough of the theory woven in to let you advance on your own if you so choose.

This CSS 101 class does have a prerequisite: You need to have at least a basic working knowledge of XHTML. A lot of working with CSS deals with redefining existing XHTML tags, so you'll need to know one tag from another in order to write CSS.

Scope of Coverage

As you might have guessed from the previous section, the level of this book is from beginner to intermediate. For example, the text doesn't get into the details and complexity of positioning, and it doesn't take time to talk about features you can't use in the browsers most commonly used now. I'll present the basic concepts of CSS in a direct manner to show how they relate to creating Web pages.

Among the topics covered in this book are the following:

- Understanding the structure of CSS rules
- Learning how CSS and XHTML work together
- Understanding selectors and using them efficiently
- Understanding inheritance and specificity
- Setting fonts, colors, and backgrounds
- Creating whitespace in a layout
- Using basic floats for page layout
- Dealing with form elements
- Styling tables
- Creating a style sheet for print

Objectives

The overriding objective of this book is to establish your core competency with CSS. To that end, this book will discuss basic techniques, teach you some fundamentals of CSS, and concentrate on applying the concepts in the real world. As mentioned, this book does not attempt to turn you into a world-class expert in CSS; doing so would require five times the pages and several years of hands-on practice.

No introductory text can make you an expert. Instead, the goal of any good introduction is to lay the groundwork for you to move in that direction after finishing the work. That's exactly what I've tried to do in this book.

So let's get to it, shall we?

NOTE:

Learning HTML

If you're looking to learn the essentials of HTML (**H**yper**T**ext **M**arkup **L**anguage), register for the free 24-hour pass to the lynda.com Online Training Library provided in the introduction of this book, and check out the following video-based training resource:

HTML Essential Training with William E. Weinman

What is CSS?

Before you invest the time needed to read this book, you should know how you can put CSS to use.

Like the Clothes on Your Back

To describe it in terms of a simple analogy, CSS is the clothes you put on a Web page, and the XHTML document is the body. When you want a new look, you just change your clothes, right? In an ideal world, a new look for a Web page would require nothing more than a new style sheet.

Back in the "old days," when the look of a page was driven by tables, spacer GIFs, sliced-up images, and background colors in table cells, it was as though your clothes were actually a surgical implant rather than a thin covering of cloth. Back then, altering the appearance of a Web page was akin to changing your look by going for reconstructive surgery!

So, the XHTML document is like your body (or, if you prefer, your **`<body>`**), and the various elements making up its structure are like your skeleton. Rather than demanding a change to the skeleton just to make a page look different, CSS lets you drape a set of clothes over that skeleton.

The CSS-as-clothes analogy holds up in another fashion: In the same way your clothes depend on your body's skeleton to take their intended shape, CSS depends on a Web page's structure in order to work properly. This doesn't mean it requires a change of structure for a change of appearance, but it does need the structure to function properly.

Simple Yet Powerful

A more geeky way of describing CSS is that it's a simple declarative language for suggesting the presentation of a Web page. This means CSS comprises straightforward declarations, such as **`h1 {color: red;}`**. Even if you've never seen a lick of CSS, as long as you know what an **`h1`** is, you can probably make a good guess as to the intended result of that statement: Any **`h1`** should be colored red.

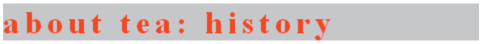
As for "suggesting the presentation," it's important to establish that CSS does not guarantee that a page will look a certain way. This might sound scary, but in all honesty, nothing has ever guaranteed the layout of a page. What's that, you say? Table layouts looked the same in all browsers? Not so! Netscape 1.0, for example, didn't support tables. It couldn't have: Netscape 1.1 was the first widely available browser to implement tables. It was also the browser to give us the **`font`** element, image alignment, and a whole host of other features. So, the first table-and-spacer designers created layouts knowing their pages would fall apart for some people.

To return to CSS, the general idea is that a style sheet offers a strong suggestion for a page's presentation, and the browser will generally go along with those suggestions unless it can't or unless it has been set to ignore some or all of them. In practice, however, hardly anyone ever tells their browser to ignore CSS.

Here's another simple example of CSS:

```
h1 {color: red; background: #CCCCCC;
letter-spacing: 5px;}
```

Again, even if you've never seen CSS before, you can probably make some intelligent guesses about the intended effect here. The color of **`h1`**s should be red; their backgrounds should be a light gray (OK, I admit you had to be able to recognize hexadecimal color notation there); and the spacing between letters should be 5 pixels.



about tea: history

The result of the simple CSS example

In each case, a simple declaration was made: the color should be red, the background should be **`#CCCCCC`**, and so on. Each one of these is simple to understand and limited in scope—but put them together, and you can arrive at some pretty sophisticated effects. (Granted, red text on a light gray background isn't exactly sophisticated, but bear with us. The book is still young.)

It's this combined complexity that makes CSS something like one of those abstract strategy board games advertised so heavily in the 1970s: "minutes to learn, but a lifetime to master." Fortunately, from personal experience, I can tell you that it takes considerably less than a lifetime to master CSS.

VIDEO:

what_is_css.mov

You can make a single Web page look radically different just by changing its style sheet, without touching a single line of the XHTML. To learn more about what CSS is, check out **what_is_css.mov** in the **videos** folder on the **CSS HOT CD-ROM**.

Why Use CSS?

Maybe the promise of complex presentation through simple declarations isn't enough for you. In that case, I have some other reasons why using CSS is a good idea.

It Makes Markup More Maintainable

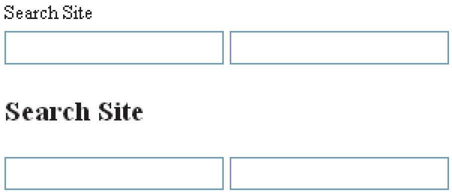
Honestly, which would you rather have to edit? This is the first snippet:

```
<table cellspacing="0" cellpadding="1">
<tr>
<td bgcolor="#332200">
<table cellspacing="0" cellpadding="2"
bgcolor="#F0E6C3">
<tr>
<td>
<font size="+2" color="#FFFFFF">Search
Site</font>
</td>
</tr>
<tr>
<form action="/scripts/search">
<td nowrap>
<input type="text" name="search-terms" />
<input type="image" src="/images/search.gif" />
</td>
</form>
</tr>
</table>
</td>
</tr>
</table>
```

And this is the second snippet:

```
<div class="portlet input-allowed">
<h3>Search Site</h3>
<form action="/scripts/search">
<input type="text" name="search-terms" />
<input type="image" src="/images/search.gif" />
</form>
</div>
```

Consider that the first snippet has invalid markup whereas the second does not, and consider that the second example can look the same as the first, if not better.



Which one was done with tables? Which one was done with CSS? Hey, if you can't tell, why should we?

It Makes Pages Smaller

In the previous section, it's pretty obvious which example uses less code and therefore would produce a smaller XHTML document. Now envision a similar degree of markup reduction throughout an entire page.

In fact, a number of studies over the years have determined that if you take a site and rework its markup using CSS for layout instead of tables, you can cut the size of the XHTML document in half. That's right, you'll get a 50 percent reduction in document size. On occasion, that reduction can be less than or greater than 50 percent, but in the majority of cases, a 50 percent reduction is possible.

It is true that some of what's lost in markup weight is gained back in increased CSS weight. However, usually the final style sheet is 25 to 35 percent of the original page's weight. So, you end up with an XHTML+CSS combination that's 15 to 25 percent smaller than the table-and-spacer XHTML document alone.

Furthermore, that 15 to 25 percent savings comes when a visitor loads their first page on your site. After that, assuming you've set up your site properly, the savings can jump to 50 percent. (You'll see how to set up your site properly in Chapter 2, *"Understanding CSS."*)

It Makes Sites Faster

CSS is a magic Web accelerator? Well, yes.

Consider this: Your visitors don't care how your pages are constructed. They care only about two issues: getting the information they want and getting it quickly. A site that takes 2 seconds for its pages to load will seem superior to one that takes 4 seconds per page. Because CSS uses less code, it makes your overall file sizes smaller, which in turn means your pages will load more quickly into your visitors' browsers.

The number-one factor in page response time is how many bytes are shipped over the wire. So if your pages are half their former size, they'll take half as long to load. That may seem like an obvious point, but it needs to be said.

It Helps with Search Engine Optimization

No, CSS is not the magic bullet that will put your site at the top of the Google rankings; we're not selling snake oil here. However, properly using CSS can help boost your site in the rankings, even if just a little bit.

The way search engines work (at least as of this writing) is that content coming at the top of a page has more weight than content appearing later in the page. In other words, the "further down" a piece of content appears in a document's source, the less relevant it's considered to be by search engines.

Thanks to CSS, you can order a document so the most important information comes sooner and the less important stuff comes later. This is true even if you visually lay out the page so the more important stuff goes on the right and the less important stuff goes on the left. (This might not be a good idea in left-to-right languages such as English, but hey, it's your design.)

NOTE:

Recognizing the Exception

The assertion here that "the number-one factor in page response time is how many bytes are shipped over the wire" does have an exception. For extremely high-traffic sites, page load times are actually shorter when the CSS is embedded in the document, even if that makes the page bigger. This is because the server resources required to handle two requests (one for the XHTML document and one for the style sheet) become significant when the servers are loaded with so much traffic.

You aren't likely to run into this situation unless you have a really high-traffic site—and when we say "really high-traffic," we mean traffic levels like those at eBay or Yahoo. So, odds are you'll never have to worry about this. Just in case, though, forewarned is forearmed.

Authoring CSS

You can create CSS in many ways. You can use anything from free text-editing applications that come with your Mac or Windows operating system to full-blown Web development programs, such as GoLive or Macromedia Dreamweaver, both from Adobe. The following sections introduce some popular applications you can use to create your style sheets.

A Plain Old Text Editor

You can write CSS in anything you can use to write XHTML. If you use Windows, you already have the free Notepad and WordPad applications that come with the operating system. If you're on a Mac, you have TextEdit. You can even use professional word-processing software such as Microsoft Word or Corel WordPerfect or even programs such as emacs, vi, joe, pico… anything you can use to write and save text files.

My personal favorite is BBEdit (**www.barebones.com**), an editor for the Mac from Bare Bones Software. I'll be using BBEdit for the majority of the examples and exercises in this book. If you're using Windows and would like to use a program similar to BBEdit, I suggest EditPlus (**www.editplus.com**) from ES-Computing. Both applications are high-powered text editors that allow for grep-style search-and-replace tasks and have features such as syntax coloring for programming languages built in. At their cores, however, they're text editors.

If text editors are a little too plain for you, some programs are specifically for CSS authoring, or at least for Web development.

Style Master

A product of Western Civilisation (**http://westciv.com**) and John Allsopp, who was one of the original CSS Samurai at the Web Standards Project, Style Master is a commercial (read: not free) CSS-authoring program available for both the Mac and Windows. Among hundreds of other features, Style Master features several design templates, a built-in tutorial, and browser compatibility information. As with many other Web design programs, Style Master has autocompletion for CSS code, a design preview mode, and the ability to work with content management systems such as blogging software.

TopStyle

TopStyle (**http://bradsoft.com/topstyle**) from NewsGator Technologies is the brainchild of Nick Bradbury, the author of HomeSite, a venerable markup editor. It's available for Windows and is promoted as a CSS/XHTML editor. Like Style Master, it offers autocompletion, compatibility checking, and a preview mode, along with hundreds of other features. Also like Style Master, TopStyle is oriented more toward hands-on authors who like to see the code as they write it instead of having it hidden behind a point-and-click interface.

Dreamweaver

Dreamweaver (**www.adobe.com**) is a complete Web development application offered by Adobe (formerly Macromedia). In many ways, Dreamweaver offers a visually oriented development space as well as the ability to dig into the code of a Web page. It offers powerful CSS-centric tools, dialog boxes that let authors fill in values, and help files to make using CSS easier.

Now that you understand what CSS is and the benefits of using it, in the next chapter, *"Understanding CSS,"* you'll learn about some of the fundamental principles of working with CSS, including understanding key terms, adding rules to a style sheet, and using selectors.