The Essential Guide to CSS and HTML Web Design

Craig Grannell

friendsof
DESIGNER TO DESIGNER™
an Apress® company

The Essential Guide to CSS and HTML Web Design

ISBN-13 (pbk): 978-1-59059-907-5

ISBN-10 (pbk): 1-59059-907-1

Printed and bound in the United States of America 9 8 7 6 5 4 3 2 1

Distributed to the book trade worldwide by Springer-Verlag New York, Inc., 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax 201-348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com.

For information on translations, please contact Apress directly at 2855 Telegraph Avenue, Suite 600, Berkeley, CA 94705. Phone 510-549-5930, fax 510-549-5939, e-mail info@apress.com, or visit www.apress.com.

The source code for this book is freely available to readers at www.friendsofed.com in the Downloads section.

Credits

Lead Editors
Chris Mills, Tom Welsh

Technical Reviewer
David Anderson

Editorial Board
Steve Anglin, Ewan Buckingham, Gary Cornell, Jonathan Gennick, Jason Gilmore, Jonathan Hassell, Matthew Moodie, Jeffrey Pepper, Ben Renow-Clarke, Dominic Shakeshaft, Matt Wade, Tom Welsh

Project Manager
Kylie Johnston

Copy Editor
Damon Larson

Assistant Production Director
Kari Brooks-Copony

Production Editor
Ellie Fountain

Compositor
Dina Quan

Artist
April Milne

Proofreader
Nancy Sixsmith

Indexer
Julie Grady

Interior and Cover Designer
Kurt Krames

Manufacturing Director
Tom Debolski

CONTENTS AT A GLANCE

CONTENTS

ABOUT THE AUTHOR

Craig Grannell is a well-known web designer and writer who's been flying the flag for web standards for a number of years. Originally trained in the fine arts, the mid-1990s saw Craig become immersed in the world of digital media, his creative projects encompassing everything from video and installation-based audio work, to strange live performances—sometimes with the aid of a computer, televisions, videos, and a PA system, and sometimes with a small bag of water above his head. His creative, playful art, which usually contained a dark, satirical edge, struck a chord with those who saw it, leading to successful appearances at a number of leading European media arts festivals.

Craig soon realized he'd actually have to make a proper living, however. Luckily, the Web caught his attention, initially as a means to promote his art via an online portfolio, but then as a creative medium in itself, and he's been working with it ever since. It was during this time that he founded Snub Communications (`www.snubcommunications.com`), a design and writing agency whose clients have since included the likes of Rebellion Developments (publishers of 2000 AD), IDG UK (publishers of *Macworld*, *PC Advisor*, *Digital Arts*, and other magazines), and Swim Records.

Along with writing the book you're holding right now, Craig has authored *Web Designer's Reference* (friends of ED, 2005) and various books on Dreamweaver, including *Foundation Web Design with Dreamweaver 8* (friends of ED, 2006). Elsewhere, he's written numerous articles for *Computer Arts*, *MacFormat*, *.net/Practical Web Design*, 4Talent, *MacUser*, the dearly departed *Cre@te Online*, and many other publications besides.

When not designing websites, Craig can usually be found hard at work in his quest for global superstardom by way of his eclectic audio project, the delights of which you can sample at `www.projectnoise.co.uk`.

ABOUT THE TECHNICAL REVIEWER

David Anderson is a biochemistry graduate from North West England who first noticed the value of the Internet in the early 1990s while using it as a research tool to aid his academic studies. He created his first website shortly after graduating in 1997, and began to establish himself as a freelance developer while also working in a variety of roles for several major UK companies until eventually founding his own business, S2R Creations, in 2003.

David discovered the web standards movement early in his career, and quickly adapted his working practices to utilize the power and versatility of CSS and semantic HTML. Clients benefiting from his skills have included New Directions Recruitment and Rex Judd Ltd. He has been sharing his knowledge with members of various web development forums for over five years, has written for *Practical Web Design* magazine, and has established his reputation as an authority on web standards as a result.

When he isn't developing websites, he can be found taking photos of anything that will stay still long enough, as well as a few things that won't. He shares his photos on Flickr, at `www.flickr.com/photos/ap4a`, and also writes on his blog at `www.ap4a.co.uk`.

ACKNOWLEDGMENTS

Writing a book is a long process, involving many hours of effort. To see the final product is exhilarating and extremely satisfying, but it couldn't have happened without those who've supported me along the way. In particular, I'd like to thank David Anderson, whose excellent editing, reviewing, ideas, and suggestions were indispensable in the revision of the text. Special thanks also to Chris Mills for getting the ball rolling, to Tom Welsh for picking up the baton, and to Kylie Johnston for keeping everything ticking over. Thanks also to the other members of the friends of ED team for their hard work in getting this publication into the world.

I'm also extremely fortunate to have had the support of several other great designers. I particularly owe a debt of gratitude to Sarah Gay (`www.stuffbysarah.net`) for her highly useful, selfless contributions, and to my former partner in crime David Powers, who once again stepped in to assist with a couple of elements in the book. Thanks also to Jon Hicks, Matthew Pennell, and Lokesh Dhakar for granting permission to include elements of their work, and to the many designers whose work has been an inspiration over the years.

And, finally, thanks to Kay for once again being there for me and putting up with me while I wrote this book.

FOREWORD

Designing for the Web is a wonderful thing. The ability to publish something and have it appear immediately and globally is an empowering feeling. I'll never forget the first rush I felt when, as a print designer, I could simply "upload" some files and have them be immediately visible, rather than waiting in trepidation for the boxes to return from the printer. Back then the Web was simpler, there were fewer materials and tools, and "styling" was something you hacked together using bizarre hacks and workarounds to achieve even the simplest of tasks. The browser landscape was equally testing.

Now we're in a much better position. We have a wonderful thing called CSS that allows us to style pages with concise style rules and leave the HTML to describe the content, not the presentation. Content can be repurposed for different media.

But anyone keen to learn web design (from scratch, or to improve their existing skills) has a bewildering job on their hands. The publishing market is saturated with good books on web design, HTML, and CSS. Yet if you were asked for a single book that encompasses all three, and that someone could understand without assuming any prior "Internet knowledge," what would you recommend? Still trying to think of one?

A regular contributor to *.net/Practical Web Design* magazine, Craig Grannell has written *The Essential Guide to CSS and HTML Web Design* for this purpose. Whether you need a reference for unmemorable code like HTML entities, or need to know what on earth HTML entities are, it's all here. Laid out in an understandable and non-patronizing manner, every aspect of creating a site is covered.

There are still many challenges to face when designing sites, but the sheer fun of it is better than ever. With this guide in your hands, more so!

Jon Hicks
Hicksdesign

INTRODUCTION

The Web is an ever-changing, evolving entity, and it's easy for a designer to get left behind. As both a designer and writer, I see a lot of books on web design, and although many are well written, few are truly integrated, modular resources that any designer can find useful in his or her day-to-day work. Most web design books concentrate on a single technology (or, commonly, a piece of software), leaving the designer to figure out how to put the pieces together.

This book is different

The Essential Guide to CSS and HTML Web Design provides a modern, integrated approach to web design. Each of the chapters looks at a specific aspect of creating a web page, such as type, working with images, creating navigation, and creating layout blocks. In each case, relevant technologies are explored in context and at the appropriate times, just as in real-world projects—for example, markup is explored along with associated CSS and JavaScript, rather than each technology being placed in separate chapters, and visual design ideas are discussed so you can get a feel for how code affects page layouts. Dozens of practical examples are provided, which you can use to further your understanding of each subject. This highly modular and integrated approach means that you can dip in and out of the book as you need to, crafting along the way a number of web page elements that you can use on countless sites in the future.

Because the entire skills gamut is covered—from foundation to advanced—this book is ideal for beginners and long-time professionals alike. If you're making your first move into standards-based web design, the "ground floor" is covered, rather than an assumption being made regarding your knowledge. However, contemporary ideas, techniques, and thinking are explored throughout, ensuring that the book is just as essential for the experienced designer wanting to work on CSS layouts, or the graphic designer who wants to discover how to create cutting-edge websites.

This book's advocacy of web standards, usability, and accessibility with a strong eye toward visual design makes it of use to technologists and designers alike, enabling everyone to build better websites. An entire chapter is devoted to browser issues, which should help ensure your sites look great, regardless of the end user's setup. And for those moments when a

particular tag or property value slips your mind, this book provides a comprehensive reference guide that includes important and relevant XHTML elements and attributes, XHTML entities, web colors, and CSS 2.1 properties and values.

Remember that you can visit the friends of ED support forums at `www.friendsofed.com/forums` to discuss aspects of this book, or just to chat with like-minded designers and developers. You can also download files associated with this book from `www.friendsofed.com`—just find the book in the friends of ED catalog located on the homepage, and then follow its link to access downloads and other associated resources.

Layout conventions

To keep this book as clear and easy to follow as possible, the following conventions are used throughout:

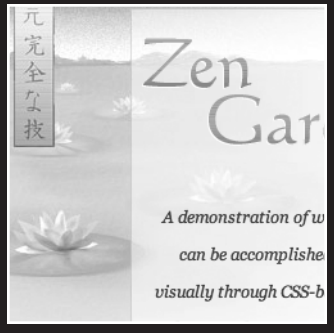
- Important words or concepts are normally highlighted on the first appearance in **bold type**.
- Code is presented in `fixed-width font`.
- New or changed code is normally presented in **`bold fixed-width font`**.
- Pseudo-code and variable input are written in *`italic fixed-width font`*.
- Menu commands are written in the form Menu ➤ Submenu ➤ Submenu.
- Where I want to draw your attention to something, I've highlighted it like this:

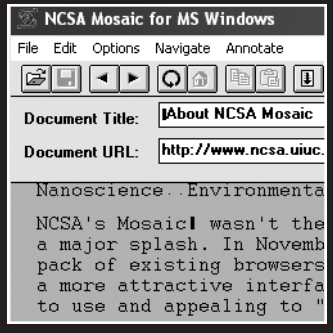
> *Ahem, don't say I didn't warn you.*

- To make it easier to work through the exercises, each one has an introductory box that lists where you can find any required files and the completed files within the downloadable file archive. A short overview of what you'll learn is also included.
- Sometimes code won't fit on a single line in a book. Where this happens, I use an arrow like this: ➥.

```
This is a very, very long section of code that should be written all on
➥ the same line without a break.
```

1 AN INTRODUCTION TO WEB DESIGN





In this chapter:

- Introducing the Internet and web design
- Working with web standards
- Working with XHTML
- Understanding and creating CSS rules
- Creating web page boilerplates
- Organizing web page content

A brief history of the Internet

Even in the wildest dreams of science fiction and fantasy writers, few envisioned anything that offers the level of potential that the Internet now provides for sharing information on a worldwide basis. For both businesses and individuals, the Internet is now the medium of choice, largely because it enables you to present your wares to the entire world on a 24/7 basis. But the technology's origins were more ominous than and very different from the ever-growing, sprawling free-for-all that exists today.

In the 1960s, the American military was experimenting with methods by which the US authorities might be able to communicate in the aftermath of a nuclear attack. The suggested solution was to replace point-to-point communication networks with one that was more akin to a net. This meant information could find its way from place to place even if certain sections of the network were destroyed. Despite the project eventually being shelved by the Pentagon, the concept itself lived on, eventually influencing a network that connected several American universities.

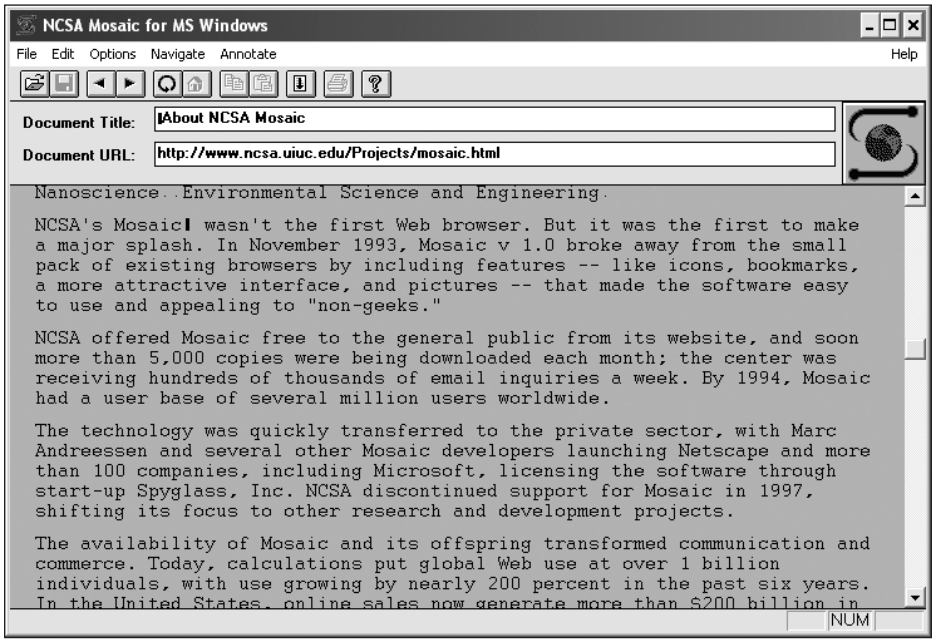
During the following decade, this fledgling network went international and began opening itself up to the general public. The term *Internet* was coined in the 1980s, which also heralded the invention of Transmission Control Protocol/Internet Protocol (TCP/IP), the networking software that makes possible communication between computers running on different systems. During the 1980s, Tim Berners-Lee was also busy working on HTML, his effort to weld hypertext to a markup language in an attempt to make communication of research between himself and his colleagues simpler.

Despite the technology's healthy level of expansion, the general public remained largely unaware of the Internet until well into the 1990s. By this time, HTML had evolved from a fairly loose set of rules—browsers having to make assumptions regarding coder intent and rendering output—to a somewhat stricter set of specifications and recommendations. This, along with a combination of inexpensive hardware, the advent of highly usable web browsers such as Mosaic (see the following image), and improved communications technology, saw an explosion of growth that continues to this day.

Initially, only the largest brands dipped their toes into these new waters, but soon thousands of companies were on the Web, enabling customers all over the globe to access information, and later to shop online. Home users soon got in on the act, once it became clear that the basics of web design weren't rocket science, and that, in a sense, everyone

could do it—all you needed was a text editor, an FTP client, and some web space. Designers soon got in on the act, increasingly catered for by new elements within HTML; Cascading Style Sheets (CSS), which took a while to be adopted by browsers, but eventually provided a means of creating highly advanced layouts for the Web; and faster web connections, which made media-rich sites accessible to the general public without forcing them to wait ages for content to download.

Therefore, unlike most media, the Web is truly a tool for everyone, and in many countries, the Internet has become ubiquitous. For those working in a related industry, it's hard to conceive that as recently as the mid-1990s relatively few people were even aware of the Internet's existence!



So, from obscure roots as a concept for military communications, the Internet has evolved into an essential tool for millions of people, enabling them to communicate with each other, research and gather information, telecommute, shop, play games, and become involved in countless other activities on a worldwide basis.

Why create a website?

Before putting pen to paper (and mouse to keyboard), it's important to think about the *reason* behind putting a site online. Millions already exist, so why do you need to create one yourself? Also, if you're working for a company, perhaps you already have plenty of marketing material, so why do you need a website as well?

I should mention here that I'm certainly not trying to put you off—far from it. Instead, I'm trying to reinforce the point that planning is key in any web design project, and although some people swear that "winging it" is the best way to go, most such projects end up gathering virtual dust online. Therefore, before doing anything else, think through why you should build a website and what you're trying to achieve.

Companies and individuals alike have practical and commercial reasons for setting up a website. A website enables you to communicate with like-minded individuals or potential clients on a worldwide basis. If you're a creative talent of some kind, you can use a website to showcase your portfolio, offering online photographs, music tracks for download, or poetry. If you fancy yourself as a journalist, a blog enables you to get your opinion out there. If you own or work for a business, creating a website is often the most efficient means of marketing your company. And even if you just have a hobby, a website can be a great way of finding others who share your passion—while you may be the only person in town who likes a particular movie or type of memorabilia, chances are there are thousands of people worldwide who think the same, and a website can bring you all together. This is perhaps why the paper fanzine has all but died, only to be reborn online, where development costs are negligible and worldwide distribution is a cinch.

In practical terms, a website exists online all day, every day (barring the odd hiccup with ISPs), which certainly isn't the case with printed media, which is there one minute and in the recycle trash the next. Distribution is less expensive than sending out printed material—a thousand-page website can be hosted for $10 per month or less, but sending a thousand-page document to one person (let alone a thousand or several thousand) may cost more than that. Likewise, development (particularly corrections and updates) is often significantly cheaper, too. For example, if you want to rework a print brochure, you have to redesign it and then reprint it. Reworking a section of a website often means swapping out a few files, which is efficient and affordable. So, for large companies and individuals alike, the ability to have relevant information online in a form that can often be updated in mere minutes, thereby keeping all interested parties up to date, is hard to resist!

Audience requirements

This book centers on the design and technology aspects of web design, but close attention must always be paid to your potential audience. It's no good forcing design ideas that result in inappropriate visuals, unusable navigation to all but the most technically minded of people, and huge download times on your site's unsuspecting visitors.

Prior to creating a site, you must ascertain what your audience wants and expects in terms of content, design, and how the site will work (by way of talking to the relevant people, and also, if your budget allows, by using surveys and focus groups). You don't have to take all of your audience's ideas into account (after all, many will be contradictory), but be mindful of common themes and ensure they're not ignored.

Technical considerations must be researched. If you're targeting designers, you can be fairly sure that a large proportion of the audience will be using monitors set to a high resolution and millions of colors, and you can design accordingly. If your site is aimed at business users, be mindful that much of your potential audience will likely be using laptops (or

older computers, for staff at the lower end of the ladder), with screen resolutions of 1024×768 or lower.

Determining the web browsers your audience members use is another important consideration. Although use of web standards (used throughout this book) is more likely to result in a highly compatible site, browser quirks still cause unforeseen problems; therefore, always check to see what browsers are popular with a site's visitors, and ensure you test in as many as you can. Sometimes you won't have access to such statistics, or you may just be after a "sanity check" regarding what's generally popular. A couple of useful places to research global web browser statistics are `www.w3schools.com/browsers/browsers_stats.asp` and `www.upsdell.com/BrowserNews/`. Note, though, that any statistics you see online are effectively guesswork and are not a definitive representation of the Web as a whole; still, they do provide a useful, sizeable sample that's often indicative of current browser trends.

Although you might be used to checking browser usage, and then, based on the results, designing for specific browsers, we'll be adhering closely to web standards throughout this book. When doing this, an "author once, work anywhere" approach is feasible, as long as you're aware of various browser quirks (many of which are explored in Chapter 9). Of course, you should still always ensure you test sites in as many browsers as possible, just to make sure everything works as intended.

Web design overview

Web design has evolved rapidly over the years. Initially, browsers were basic, and early versions of HTML were fairly limited in what they enabled designers to do. Therefore, many older sites on the Web are plain in appearance. Additionally, the Web was originally largely a technical repository, hence the boring layouts of many sites in the mid 1990s—after all, statistics, documentation, and papers rarely need to be jazzed up, and the audience didn't demand such things anyway.

As with any medium finding its feet, things soon changed, especially once the general public flocked to the Web. It was no longer enough for websites to be text-based information repositories. Users craved—demanded, even—color! Images! Excitement! Animation! Interaction! Even video and audio managed to get a foothold as compression techniques improved and connection speeds increased.

The danger of eye candy became all too apparent as the turn of the century approached: every site, it seemed, had a Flash intro, and the phrase "skip intro" became so common that it eventually spawned a parody website.

These days, site design has tended toward being more restrained, as designers have become more comfortable with using specific types of technologies for relevant and appropriate purposes. Therefore, you'll find beautifully designed XHTML- and CSS-based sites sitting alongside highly animated Flash efforts.

Of late, special emphasis is being placed on **usability** and **accessibility**, and, in the majority of cases, designers have cottoned to the fact that content must take precedence. However,

just because web standards, usability, and accessibility are key, that doesn't mean design should be thrown out the window. As we'll see in later chapters, web standards do not have to come at the expense of good design—far from it. In fact, a strong understanding of web standards helps to improve websites, making it easier for you to create cutting-edge layouts that work across platforms and are easy to update. It also provides you with a method of catering for obsolete devices.

If you're relatively new to web design, you may be wondering about the best platform and software for creating websites. Ultimately, it matters little which platform you choose, as long as you have access to the most popular browsers for testing purposes (a list that I'd now include Apple's Safari in, alongside Internet Explorer, Firefox, and Opera). Regarding software, there's an overview in Appendix E ("Browsers Guide"), but this isn't an exhaustive guide, so do your own research and find software to your liking.

Why WYSIWYG tools aren't used in this book

With lots of software available and this book being design-oriented, you might wonder why I'm not using WYSIWYG web design tools. This isn't because I shun such tools—it's more that in order to best learn how to do something, you need to start from scratch, with the foundations. Many web design applications make it tempting to "hide" the underlying code from you, and most users end up relying on the graphical interface. This is fine until something goes wrong and you don't know how to fix it.

Removing software from the equation also means we concentrate on the underlying technology that drives web pages, without the distraction of working out which button does what. It also ensures that the book will be relevant to you, regardless of what software you use or your current skill level. Therefore, I suggest you install a quality text editor to work through the exercises, or set your web design application to use its code view. Once you're familiar with the concepts outlined in this book, you can apply them to your work, whatever your chosen application for web design. This level of flexibility is important, because you never know when you might have to switch applications—something that's relatively painless if you know how to design for the Web and understand technologies like CSS and HTML.

Introducing HTML and XHTML

The foundation of the majority of web pages is **HyperText Markup Language**, commonly known by its initials, HTML. A curious facet of the language is that it's easy to pick up the basics—anyone who's computer literate should be able to piece together a basic page after learning some tags—but it has enough flexibility and scope to keep designers interested and experimenting, especially when HTML is combined with Cascading Style Sheets (CSS), which we'll discuss later in this chapter. This section presents an overview of HTML tags and elements, and how HTML and XHTML relate to web standards.

Introducing the concept of HTML tags and elements

HTML documents are text files that contain tags, which are used to mark up HTML elements. These documents are usually saved with the `.html` file extension, although some prefer `.htm`, which is a holdover from DOS file name limitations, which restricted you to eight characters for the file name and three for the extension.

The aforementioned tags are what web browsers use to display pages, and assuming the browser is well behaved (most modern ones are), the display should conform to standards as laid out by the **World Wide Web Consortium (W3C)**, the organization that develops guidelines and specifications for many web technologies.

> *The W3C website is found at* `www.w3.org`. *The site offers numerous useful tools, including validation services against which you can check your web pages.*

HTML tags are surrounded by angle brackets—for instance, `<p>` is a paragraph start tag. It's good practice to close tags once the element content or intended display effect concludes, and this is done with an end tag. End tags are identical to the opening start tags, but with an added forward slash: `/`. A complete HTML element looks like this:

```
<p>Here is a paragraph.</p>
```

This element consists of the following:

- **Start tag**: `<p>`
- **Content**: `Here is a paragraph.`
- **End tag**: `</p>`

> *HTML doesn't have a hard-and-fast rule regarding the case of tags, unlike XHTML, which we'll shortly be talking about and which will be used throughout the book. If you look at the source code of HTML pages on the Web, you may see lowercase tags, uppercase tags or, in the case of pages put together over a period of time, a mixture of the two. That said, it's still good practice with any markup language to be consistent, regardless of whether the rules are more flexible.*

Nesting tags

There are many occasions when tags must be placed inside each other; this process is called nesting. One reason for nesting is to apply basic styles to text-based elements. Earlier, you saw the code for a paragraph element. We can now make the text bold by surrounding the element content with a `strong` element:

```
<p><strong>Here is a paragraph.</strong></p>
```

> *You might be used to using the* `bold` *element to make text bold, but it is a* ***physical*** *element that only amends the look of text rather than also conveying semantic meaning.* ***Logical*** *elements, such as* `strong`*, convey meaning and add styling to text and are therefore preferred. These will be covered in Chapter 3.*

Note that the `strong` tags are nested within the paragraph tags (`<p></p>`), not the other way around. That's because the paragraph is the parent element to which formatting is being applied. The paragraph could be made bold *and* italic by adding another element, emphasis (`<em></em>`), as follows:

```
<p><strong><em>Here is a paragraph.</em></strong></p>
```

In this case, the `strong` and `em` tags could be in the opposite order, as they're at the same level in the hierarchy. However, you must always close nested tags in the reverse order to that in which they're opened, as shown in the previous code block, otherwise some browsers may not display your work as intended. For instance, the following should be avoided:

```
<p><strong><em>Here is a paragraph.</strong></em></p>
```

As previously mentioned, it's good practice to close tags in HTML—even though it's not a requirement for all elements, being sloppy in this area can lead to errors. Take a look at the following:

```
<p><strong><em>Here is a paragraph.</strong></p>
```

Here, the emphasis element isn't closed, meaning subsequent text-based content on the page is likely to be displayed in italics—so take care to close all your tags.

Web standards and XHTML

As mentioned earlier, we'll be working with **Extensible HyperText Markup Language (XHTML)** rules in this book, rather than HTML. The differences between HTML and XHTML are few, but important, and largely came about because of the inconsistent way that browsers displayed HTML. XHTML is stricter than HTML and has additional rules; oddly, this actually makes it easier to learn, because you don't have to worry about things like which case to use for tags and whether they require closing. You have hard-and-fast rules in each case. XHTML-specific rules are as follows.

All tags and attribute names must be in *lowercase* and must *always* be closed. Therefore, the following is incorrect:

```
<P>This is a paragraph.
<P>This is another paragraph.
```

The preceding lines should be written like this:

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Unlike HTML, all XHTML elements require an end tag, including empty elements (such as `br`, `img`, and `hr`). The HTML for a carriage return is `br`. In XHTML, this must be written `<br></br>` or, more usually, in a combination form that looks like this: `<br />`. The trailing slash is placed at the end of the start tag, with a space prior to it (now typical practice, this was initially done to ensure compatibility with aging browsers that would otherwise ignore the tag entirely if the space wasn't present).

Tags often have **attributes** that modify them in some way. For instance, two attributes for the table cell tag `td` are `nowrap` (to stop content wrapping) and `colspan` (which states how many columns this cell should span). In XHTML, attributes must be quoted and always have a value. If necessary, the attribute name itself is repeated for the value. Therefore, the following is *incorrect*:

```
<td colspan=2 nowrap>
```

Instead, in XHTML, we write this:

```
<td colspan="2" nowrap="nowrap">
```

Evolution is another aspect that we have to deal with. Just as the survival of the fittest removes some species from nature, so too are tags (and attributes) unceremoniously dumped from the W3C specifications. Such tags and attributes are referred to as **deprecated**, meaning they are marked for removal from the standard and may not be supported in future browsers. In cases when deprecated tags are used in this book, this will be highlighted (and likewise in the reference section); in most cases, these tags can be avoided.

Semantic markup

In the previous few subsections, you may have noticed specific elements being used for specific things. This is referred to as **semantic markup** and is a very important aspect of modern web design. Plenty of (X)HTML elements exist, and each one has a clearly defined purpose (although some have more than one use). Because of the flexibility of markup languages, it's often possible to "wrongly" use elements, bashing your page into shape by using elements for design tasks they're not strictly suited for and certainly weren't originally designed for.

During the course of this book, we'll talk about semantics a fair amount. Ultimately, good semantic design enables you to simplify your markup and also provides the greatest scope for being able to style it with CSS (see the following section). By thinking a little before you code and defining your content with the correct markup, you'll end up with cleaner code and make it much easier for yourself in the long run when it comes to adding presentation to your content.

Introducing CSS

CSS is the W3C standard for defining the visual presentation for web pages. HTML was designed as a structural markup language, but the demands of users and designers encouraged browser manufacturers to support and develop presentation-oriented tags. These tags "polluted" HTML, pushing the language toward one of decorative style rather than logical structure. Its increasing complexity made life hard for web designers, and source code began to balloon for even basic presentation-oriented tasks. Along with creating needlessly large HTML files, things like font tags created web pages that weren't consistent across browsers and platforms, and styles had to be applied to individual elements—a time-consuming process.

The concept behind CSS was simple, yet revolutionary: remove the presentation and separate design from content. Let HTML (and later XHTML) deal with structure, and use a separate CSS document for the application of visual presentation.

The idea caught on, albeit slowly. The initial problem was browser support. At first, most browsers supported only a small amount of the CSS standard—and badly at that. But Internet Explorer 5 for Mac made great strides with regard to CSS support, and it was soon joined by other browsers fighting for the crown of standards king. These days, every up-to-date browser supports the majority of commonly used CSS properties and values, and more besides.

Another problem has been educating designers and encouraging them to switch from old to new methods. Benefits constantly need to be outlined and proven, and the new methods taught. Most designers these days style text with CSS, but many still don't use CSS for entire web page layouts, despite the inherent advantages in doing so. This, of course, is one of the reasons for this book: to show you, the designer, how CSS can be beneficial to you—saving you (and your clients) time and money—and to provide examples for various areas of web page design and development that you can use in your sites.

In this section we'll look at separating content from design, CSS rules, CSS selectors and how to use them, and how to add styles to a web page.

Separating content from design

Do you ever do any of the following?

- Use tables for website layout
- Use invisible GIFs to "push" elements around your web page
- Hack Photoshop documents to bits and stitch them back together in a web page to create navigation elements and more
- Get frustrated when any combination of the previous leads to unwieldy web pages that are a pain to edit

If so, the idea of separating content from design should appeal to you. On one hand, you have your HTML documents, which house content marked up in a logical and semantic manner. On the other hand, you have your CSS documents, giving you site-wide control of

the presentation of your web page elements from a single source. Instead of messing around with stretching transparent GIFs, and combining and splitting table cells, you can edit CSS rules to amend the look of your site, which is great for not only those times when things just need subtle tweaking, but also when you decide everything needs a visual overhaul. After all, if presentation is taken care of externally, you can often just replace the CSS to provide your site with a totally new design.

Designers (and clients paying for their time) aren't the only ones to benefit from CSS. Visitors will, too, in terms of faster download times, but also with regard to **accessibility**. For instance, people with poor vision often use screen readers to surf the Web. If a site's layout is composed of complex nested tables, it might visually make sense; however, the underlying structure may not be logical. View the source of a document and look at the order of the content. A screen reader reads from the top to the bottom of the *code* and doesn't care what the page looks like in a visual web browser. Therefore, if the code compromises the logical order of the content (as complex tables often do), the site is compromised for all those using screen readers.

Accessibility is now very important in the field of web design. Legislation is regularly passed to strongly encourage designers to make sites accessible for web users with disabilities. It's likely that this trend will continue, encompassing just about everything except personal web pages. (However, even personal websites shouldn't be inaccessible.)

The rules of CSS

Style sheets consist of a number of **rules** that define how various web page elements should be displayed. Although sometimes bewildering to newcomers, CSS rules are simple to break down. Each rule consists of a **selector** and a **declaration**. The selector begins a CSS rule and specifies which part of the HTML document the rule will be applied to. The declaration consists of a number of property/value pairs that set specific properties and determine how the relevant element will look. In the following example, p is the selector and everything thereafter is the declaration:

```
p {
  color: blue;
}
```

As you probably know, p is the HTML tag for a paragraph. Therefore, if we attach this rule to a web page (see the section "Adding styles to a web page" later on in this chapter for how to do so), the declaration will be applied to any HTML marked up as a paragraph, thereby setting the color of said paragraphs to blue.

> *CSS property names are not case sensitive, but it's good to be consistent in web design—it's highly recommended to always use lowercase. Note, though, that XML is case sensitive, so when using CSS with XHTML documents served with the proper XHTML MIME type, everything must be consistent. Also, the W3 specifications recommend that CSS style sheets for XHTML should use lowercase element and attribute names.*