**Organizing a Page with Paragraphs and Line Breaks**

When a web browser displays HTML pages, it pays no attention to line endings or the number of spaces between words.

Listing 2.2 HTML Containing Paragraph and Line Breaks

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"

"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">

<head>

<title>The Advertising Agency Song</title>

</head>

<body>

<p>

When your client's hopping mad, put his picture in the ad. If he still should prove refractory, add a picture of his factory.

</p>

<hr />

<p>

When your client's hopping mad,<br /> put his picture in the ad.

</p>

<p>

If he still should prove refractory,<br /> add a picture of his factory.

</p>

</body>

</html>

You must use HTML tags if you want to control where line and paragraph breaks actually appear. When text is enclosed within the <p></p> container tags, a line break will be assumed after the closing tag. In later chapters, you will learn to control the height of the line break using CSS. The <br /> tag forces a line break within a paragraph. Unlike the other tags you’ve seen so far, <br /> doesn’t require a closing </br> tag—this is one of those empty tags discussed earlier. Although HTML 4 does not require the / in empty tags, XHTML does and future standards will, so it’s important for you to stick to the latest standards and create web pages that are coded properly. Always code empty tags so that they end with />.

**Caution**

You might come across a lot of web content that includes <br> instead of <br />. Or you might see other content that does not include the closing </p> tag. Just remember there is a lot of antiquated web content floating around the Internet, and just because you see it in use doesn’t mean it’s correct. Save yourself a lot of future work and frustration by adhering to the standards you learn in this book. Developing clean HTML coding habits is a very important part of becoming a successful web designer.

The poem in Listing 2.2 shows the <br /> and <p> tags being used to separate the lines and verses of an advertising agency song. You might have also noticed the <hr /> tag in the listing, which causes a horizontal rule line to appear on the page (see Figure 2.2). Inserting a horizontal rule with the <hr /> tag also causes a line break, even if you don’t include a <br /> tag along with it. Like <br />, the <hr/> horizontal rule tag is an empty tag and therefore never gets a closing </hr> tag.

**Try It Yourself: Formatting Text in HTML**

Take a passage of text and try your hand at formatting it as proper HTML.

**1.** Add <html><head><title>My Title</title></head><body> to the beginning of the text (using your own title for your page instead of My Title). Also include the boilerplate code at the top of the page that takes care of meeting the requirements of XHTML.

**2.** Add </body></html> to the very end of the text.

**3.** Add a <p> tag at the beginning of each paragraph and a </p> tag at the end of each paragraph.

**4.** Use <br /> tags anywhere you want single-spaced line breaks.

**5.** Use <hr /> to draw horizontal rules separating major sections of text, or wherever you’d like to see a line across the page.

**6.** Save the file as mypage.html (using your own filename instead of mypage).

**7.** Open the file in a web browser to see your web content. (Send the file via FTP to your web hosting account, if you have one.)

**8.** If something doesn’t look right, go back to the text editor to make corrections and save the file again (and send it to your web hosting account, if applicable). You then need to click Reload/Refresh in the browser to see the changes you made.

**Caution**

If you are using a word processor to create the web page, be sure to save the HTML file in plaintext or ASCII format.

**Organizing Your Content with Headings**

When you browse through web pages on the Internet, you’ll notice that many of them have a heading at the top that appears larger and bolder than the rest of the text. Listing 2.3 is sample code and text for a simple web page containing an example of a heading as compared to normal paragraph text. Any text between <h1> and </h1> tags will appear as a large heading. Additionally, <h2> and <h3> make progressively smaller headings, and so on as far down as <h6>.

Listing 2.3 Heading Tags

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"

"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">

<head>

<title>My Widgets</title>

</head>

<body>

<h1>My Widgets</h1>

<p>My widgets are the best in the land. Continue reading to

learn more about my widgets.</p>

<h2>Widget Features</h2>

<p>If I had any features to discuss, you can bet I'd do

it here.</p>

<h3>Pricing</h3>

<p>Here, I would talk about my widget pricing.</p>

<h3>Comparisons</h3>

<p>Here, I would talk about how my widgets compare to my

competitor's widgets.</p>

</body>

</html>

**Note**

By now you’ve probably caught on to the fact that HTML code is often indented by its author to reveal the relationship between different parts of the HTML document. This indentation is entirely voluntary— you could just as easily run all the tags together with no spaces or line breaks and they would still look fine when viewed in a browser. The indentations are for you so that you can quickly look at a page full of code and understand how it fits together. Indenting your code is a very good web design habit and ultimately makes your pages easier to maintain.

As you can see in Figure 2.3, the HTML that creates headings couldn’t be simpler. In this example, the phrase “My Widgets” is prominently displayed using the <h1> tag. To create the biggest (level 1) heading, just put an <h1> tag at the beginning and a </h1> tag at the end of the text you want to use as a heading. For a slightly smaller (level 2) heading—for information that is of lesser importance than the title—use the <h2> and </h2> tags around your text. For content that should appear even less prominently than a level 2 heading, use the <h3> and </h3> tags around your text.

**Figure 2.3** The use of three levels of headings shows the hierarchy of content on this sample product page. However, bear in mind that your headings should follow a content hierarchy; use only one level 1 heading, have one (or more) level 2 headings after the level 1 heading, use level 3 headings directly after level 2 headings, and so on. Do not fall into the trap of assigning headings to content just to make that content display a certain way. Instead, ensure that you are categorizing your content appropriately (as a main heading, a secondary heading, and so on), while using display styles to make that text render a particular way in a web browser.

Theoretically, you can also use <h4>, <h5>, and <h6> tags to make progressively less important headings, but these aren’t used very often. Web browsers seldom show a noticeable difference between these headings and the <h3> headings anyway, and content usually isn’t displayed in such a manner as to need six levels of headings to show the content hierarchy.

**Note**

On many web pages nowadays, graphical images of ornately rendered letters and logos are often used in place of the ordinary text headings discussed in this chapter. However, using text headings is one of many search engine optimization (SEO) tips that you will learn about in Chapter 28, “Helping People Find Your Web Pages.” Search engines look at heading tags to see how you organize your content; they give higher preference to content that you have indicated is more important (for example, a level 1 heading) versus content that you indicate is of lesser importance (lower-level headings).

It’s important to remember the difference between a *title* and a *heading*. These two words are often interchangeable in day-to-day English, but when you’re talking HTML, <title> gives the entire page an identifying name that isn’t displayed on the page itself; it’s displayed only on the browser window’s title bar. The heading tags, on the other hand, cause some text on the page to be displayed with visual emphasis. There can be only one <title> per page and it must appear within the <head> and </head> tags, whereas you can have as many <h1>, <h2>, and <h3> headings as you want, in any order that suits your fancy.

However, as I mentioned before, you should use the heading tags to keep tight control over content hierarchy; do not use headings as a way to achieve a particular look because that’s what CSS is for.

You’ll learn to take complete control over the appearance of text on your web pages in Parts II and III of this book. Short of taking exacting control of the size, family, and color of fonts, headings provide the easiest and most popular way to draw extra attention to important text.

**Caution**

Don’t forget that anything placed in the head of a web page is not intended to be viewed on the page, whereas everything in the body of the page is intended for viewing.

**Validating Your Web Content**

In the first chapter, I discussed ways to test your pages; one very important way to test your pages is to *validate* them. Think of it this way: It’s one thing to design and draw a beautiful set of house plans, but it’s quite another for an architect to stamp it as a safe structure suitable for construction.

Validating your web pages is a similar process; in this case, however, the architect is an application —not a person.

In brief, validation is the process of testing your pages with a special application that searches for errors and makes sure your pages follow the strict XHTML standard. Validation is simple. In fact, the standards body responsible for developing web standards—the World Wide Web Consortium (W3C)

—offers an online validation tool you can use. To validate a page, follow this URL:

<http://validator.w3.org/> The W3C Markup Validation Service enables you to validate an HTML (XHTML) document to ensure it has been coded accurately.

**Peeking at Other Designers’ Pages**

Given the visual and sometimes audio pizzazz present in many popular web pages, you probably realize that the simple pages described in this chapter are only the tip of the HTML iceberg. Now that you know the basics, you might surprise yourself with how much of the rest you can pick up just by looking at other people’s pages on the Internet.

You can see the HTML for any page by right-clicking and selecting View Source in any web browser.

Don’t worry if you aren’t yet able to decipher what some HTML tags do or exactly how to use them yourself. You’ll find out about all those things in the next few chapters. However, sneaking a preview now will show you the tags that you do know in action and give you a taste of what you’ll soon be able to do with your web pages.

If you’ve already published a page online, you can use the Validate by URI tab. Use the Validate by File Upload tab to validate files stored on your local computer file system. The Validate by Direct Input tab enables you to paste the contents of a file from your text editor. If all goes well, your page will get a passing report.

If a page passes the W3C Markup Validation Service, you know it is ready for prime time.

If the W3C Markup Validation Service encounters an error in your web page, it will provide specific details (including the line numbers of the offending code). This is a great way to hunt down problems and rid your pages of buggy code. Validation not only informs you whether your pages are constructed properly, it also assists you in finding and fixing problems before you post pages for the world to see.

**Tip**

Some web development tools include built-in validation features you can use in lieu of the W3C Markup Validation Service. Some examples include browser extensions such as Firebug(<http://getfirebug.com/>) and HTML Validator (http://users.skynet.be/mgueury/mozilla/), but many other programs offer similar functionality; check your user documentation.

**The Scoop on HTML, XML, XHTML, and HTML5**

In its early days, HTML was great because it allowed scientists to share information over the Internet in an efficient and relatively structured manner. It wasn’t until later that graphical web browsers were created and HTML started being used to code more than scientific papers. HTML quickly went from a tidy little markup language for researchers to an online publishing language.

After it was established that HTML could be jazzed up for graphical browsing, the creators of web browsers went crazy by adding lots of nifty features to the language. Although these new features were neat at first, they compromised the original design of HTML and introduced inconsistencies when it came to how browsers displayed web pages; new features worked on only one browser or another, and you were out of luck if you happened to be running the wrong browser.

HTML started to resemble a bad remodelling job of a house—a job done by too many contractors and without proper planning. As it turns out, some of the browser-specific features created during this time have now been adopted as standards whereas others have been dropped completely.

As with most revolutions, the birth of the Web was very chaotic, and the modifications to HTML reflected that chaos. Over the years, a significant effort has been made to reel in the inconsistencies of HTML and restore some order to the language. The problem with disorder in HTML is that it results in web browsers having to guess at how a page is to be displayed, which is not a good thing.

Ideally, a web page designer should be able to define exactly how a page is to look and have it look the same regardless of what kind of browser or operating system someone is using. Better still, a designer should be able to define exactly what a page *means* and have that page look consistent across different browsers and platforms. This utopia is still off in the future somewhere, but a markup language called XML (Extensible Markup Language) began to play a significant role in leading us toward it.

*XML* is a general language used to create specific languages, such as HTML. It might sound a little strange, but it really just means that XML provides a basic structure and set of rules to which any markup language must adhere.

Using XML, you can create a unique markup language to describe just about any kind of information, including web pages. Knowing that XML is a language for creating other markup languages, you could create your own version of HTML using XML. You could even create a markup language called BCCML (Bottle Cap Collection Markup Language), for example, which you could use to create and manage your extensive collection of rare bottle caps.

The point is that XML lays the ground rules for organizing information in a consistent manner, and that information can be anything from web pages to bottle caps.

You might be thinking that bottle caps don’t have anything to do with the Web, so why mention them?

The reason is that XML is not entirely about web pages. XML is actually broader than the Web in that it can be used to represent any kind of information on any kind of computer. If you can visualize all the information whizzing around the globe among computers, mobile phones, handheld computers, televisions, and radios, you can start to understand why XML has much broader applications than just cleaning up web pages. However, one of the first applications of XML is to restore some order to theWeb, which is why XML is relevant to learning HTML. If XML describes data better than HTML, does it mean that XML is set to upstage HTML as the markup language of choice for the Web? No. XML is not a replacement for HTML; it’s not even a competitor of HTML. XML’s impact on HTML has to do with cleaning up HTML. HTML is a relatively unstructured language that benefits from the rules of XML.

The natural merger of the two technologies resulted in HTML’s adherence to the rules and structure of XML. To accomplish this merger, a new version of HTML was formulated that follows the stricter rules of XML. The new XML-compliant version of HTML is known as XHTML. Fortunately for you, you’ll actually be learning XHTML throughout this book because it is really just a cleaner version of HTML.

You might have heard about HTML5, which is touted as the next web standard. It will be, but not quite yet. When it does become a web standard, it will not render XHTML useless—HTML5 is not a replacement for XHTML, but instead is a major revision of HTML 4. In other words, XHTML and HTML5 can coexist on the Web, and web browsers that currently support XHTML will also (one day) support HTML5 as well.

The goal of this book is to guide you through the basics of web publishing, using XHTML and CSS as the core languages of those pages. However, whenever possible, I will note elements of the languages that are not present in HTML5, should you want to design your content for even further sustainability.

If you gain a solid understanding of web publishing and the ways in which CSS works with the overall markup language of the page (be it XHTML or HTML5), you will be in a good position if you decide you want to move from XHTML to HTML5.

**Summary**

This chapter introduced the basics of what web pages are and how they work, including the history and differences between HTML and XHTML. You learned that coded HTML commands are included in a text file, and that typing HTML text yourself is better than using a graphical editor to create HTML commands for you—especially when you’re learning HTML.

You were introduced to the most basic and important HTML tags. By adding these coded commands to any plain-text document, you can quickly transform it into a bona fide web page. You learned that the first step in creating a web page is to put a few obligatory HTML tags at the beginning and end, including a title for the page. You then mark where paragraphs and lines end and add horizontal rules and headings if you want them.