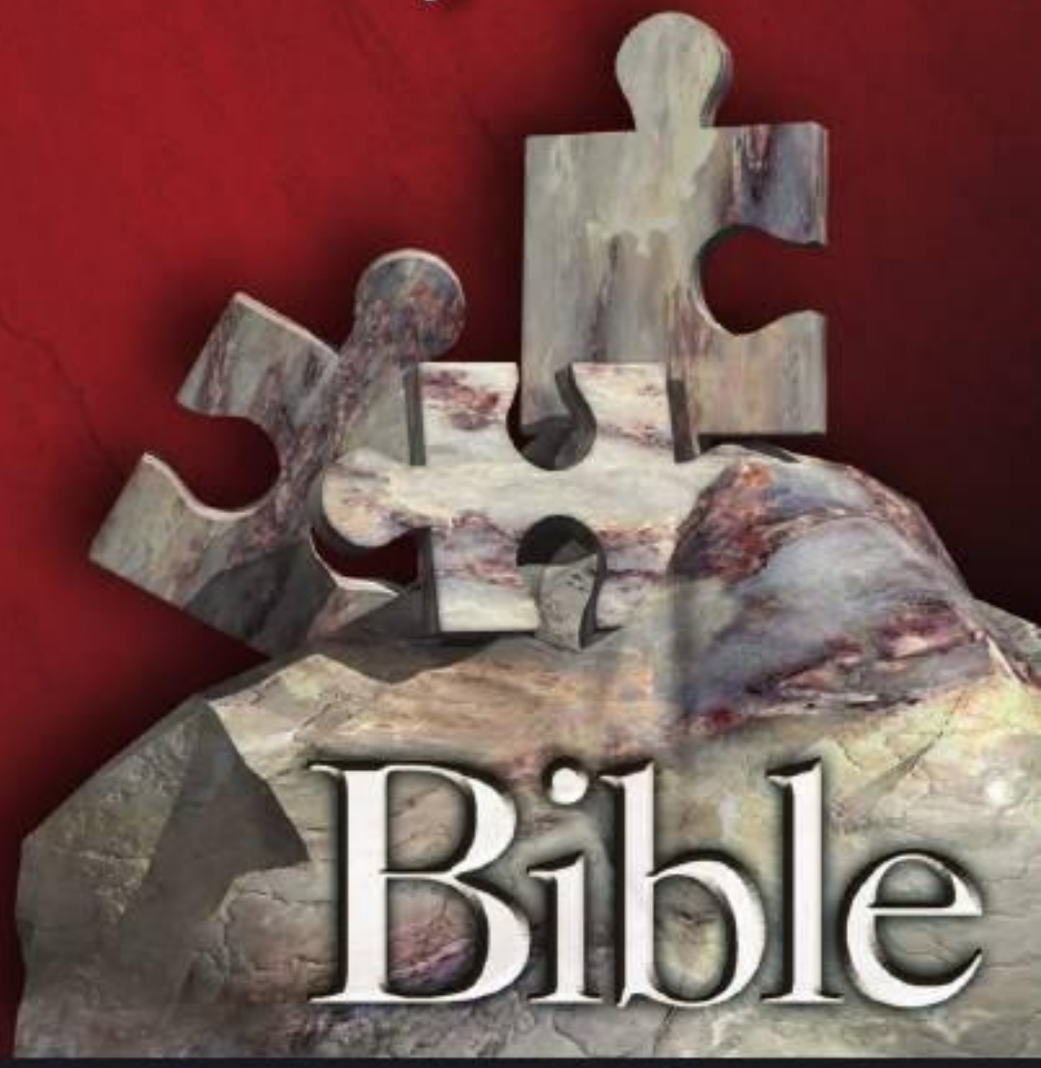
Companion Web Site

Steven M. Schafer

HTML, XHTML, and CSS

Fifth Edition



Master HTML 4.01, CSS 2.1, and XHTML 1.1

Construct static and dynamic Web pages

Build for a mobile and social networking world

Bible

The book you need to succeed!

HTML, XHTML, and CSS Bible

Fifth Edition

HTML, XHTML, AND CSS BIBLE

Fifth Edition

Steven M. Schafer

WILEY
Wiley Publishing, Inc.

HTML, XHTML, and CSS Bible, Fifth Edition

Published by
Wiley Publishing, Inc.
10475 Crosspoint Boulevard
Indianapolis, IN 46256
www.wiley.com

Published simultaneously in Canada

ISBN: 978-0-470-52396-4

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

Library of Congress Control Number: 2009940878

For my good friend RD, for being just that.

Also for Mother, whose faith in her son never wavered.
We all love and miss you.

About the Author

Steven M. Schafer is a broad technologist and a veteran of publishing. He's been in and around technology as a programmer, an editor, a product specialist, a technical manager, and a Web developer. Steve employs both open-source and proprietary technologies and has worked with the Internet since the mid-1990s. He can be reached by e-mail at sschafer@synergy-tech.com.

Credits

Executive Editor
Carol Long

Senior Project Editor
Adaobi Obi Tulton

Technical Editor
Shawn Kessel

Production Editor
Rebecca Anderson

Copy Editor
Luann Rouff

Editorial Director
Robyn B. Siesky

Editorial Manager
Mary Beth Wakefield

Marketing Manager
Ashley Zurcher

Production Manager
Tim Tate

Vice President and Executive Group Publisher
Richard Swadley

Vice President and Executive Publisher
Barry Pruett

Associate Publisher
Jim Minatel

Project Coordinator, Cover
Lynsey Stanford

Proofreaders
Scott Klemp and Sheilah Ledwidge, Word One

Indexer
Robert Swanson

Cover Image
Joyce Haughey

Cover Designer
Michael E. Trent

Acknowledgments

A book like this requires a lot of hard work from a lot of talented people. This talent and hard work deserves recognition and thanks. As such, the author would like to thank the following:

The management team at Wiley Publishing for continuing to support large, tutorial reference books so folks like you (the reader) can benefit.

Jenny Watson, acquisitions editor, for getting the ball rolling on this edition.

Carol A. Long, executive acquisitions editor, for picking up the rolling ball, keeping the project on track, and keeping this author both responsible and happy.

Adaobi Obi Tulton, senior project editor and the best developer I've had the pleasure to work with. Adaobi simultaneously kept everything on track and provided crucial insights and feedback throughout the process — all while treating everyone else as valuable team members.

Shawn Kessel, technical editor, for ensuring that the information provided is accurate as well as pertinent, and for providing many useful insights.

Rebecca Anderson, production editor, for making sure each element was ushered through the editing process, maintaining the high quality of writing found within.

Luann Rouff, copy editor, who proved to be another in the "best of" class. Luann helped make my gibberish legible, ironed out the organizational rough spots, and otherwise finely crafted the text.

Wiley's production department, for ensuring that the text was consistent, well organized, and legible, and made it through the production process intact.

John Daily, friend and markup guru, whose hard work is still paying dividends today.

Contents at a Glance

Contents

Introduction

Welcome to *HTML, XHTML, and CSS Bible, Fifth Edition*. This book was conceived, designed, and written to provide a comprehensive overview of the two largest Web technologies, HyperText Markup Language (HTML) and Cascading Style Sheets (CSS).

This book serves as an introduction and reference to the information you need to create documents — simple and complex — for the World Wide Web.

A Brief History of the Internet and the World Wide Web

The World Wide Web is omnipresent in our lives today, and most computers and computerized devices are connected to it. However, the Web and its underlying Internet infrastructure had a very different childhood that betrays the consumer and commercial base it has today.

The Internet has its roots in the U.S. Department of Defense Advanced Research Project Agency (ARPA) project begun in or around 1960. Among the project's goals was the ability to network computers quickly and across great distances. The network was to be designed to be almost fail-safe, enabling connected computers to continue communicating even if assorted routes between them were to fail.

In 1969, the ARPANet was born, connecting several key universities. The network continued to grow, with more and more universities coming online. One of the goals of the initial project — robust, nearly fail-safe performance — was realized via the Internet Protocol (IP). This protocol enabled communication packets to find various routes to a destination in case one or more of the routes became unstable. This communication protocol became the backbone of today's Internet, and is how the Internet got its name.

The Transmission Control Protocol was joined with the IP to provide a robust transmission suite, a marriage of two protocols to offer more flexibility and the ability to create better communications applications for the Internet.

In the 1980s, the Internet went through several transitions. Although it was highly populated by educational institutions, the U.S. military hadn't forgotten its original project. Other government agencies also took notice and joined the crowd online; and the military decided to create its own network, MILNET, lessening the load slightly.

By 1992, the Internet was far and away the most popular network in the world. During this time, Tim Berners-Lee, a British software engineer and computer scientist, created HyperText Markup Language to create documents, a protocol — HyperText Transfer Protocol (HTTP) — to

send such documents, and the first browser editor, called the World Wide Web. The "Web" soon came to the attention of the National Center for Supercomputing Applications (NCSA), where a programming team decided to create a better browser. Thus was Mosaic born, the first browser to support a high degree of multimedia. Mosaic helped usher in the crop of modern browsers we use today.

As the Web continued to be adopted outside of the government and educational sectors, it became more consumer-savvy. Many companies began using the Web infrastructure for marketing and support purposes, while many Web developers began to target a wider, nontechnical, audience.

By the early 2000s, the Web was accessible by almost any network-connected computer, many electronic devices, and some unlikely consumer devices such as automobiles. Each of these connected devices uses the same type of connection, the same languages to define documents, and the same protocols to send the information.

As more and more nontechnical users began using the Web, web "pages" began to look more like high-quality printed documents — resembling newspapers, brochures, magazines, and the like. This movement in content signaled how far the Web had come from its inception — from technical, text-only pages to full-color, heavily designed documents.

During the entire evolution of the World Wide Web, and especially in the last few years, standards, tools, and related applications have changed and evolved, sometimes at a very rapid pace. This gives Internet books a wide realm to cover.

What This Book Covers

What exactly is covered in this book? The easy answer is HTML and CSS, just as the title suggests; but with four plus notable versions of HTML, three plus notable versions of CSS, and a bevy of connected technologies, the answer is not so cut and dried.

The more exact answer is as follows:

- HTML 4.01/XHTML 1.1
- CSS 2.1
- JavaScript
- A few supporting applications to create and troubleshoot Web documents
- A few multimedia formats (graphics, video, and so on) and supporting applications

The following sections explain how these diverse sets of applications converge.

HTML 4.01/XHTML 1.1

HTML 4.01 is the latest version of HTML. This version is very stable, having been released in December 1999. Although HTML version 5 (HTML5) is in draft stage as of this writing, the specification is probably a good year (or so) away from actual release.

Note, however, that this book promotes and uses XHTML 1.1 standards. This includes standards such as the following:

- Every tag needs to be explicitly closed, whether by a matching closing tag or a slash at the end of a tag (if it has no matching closing tag).
- Every tag must be in lowercase; in other words, use `<p>` instead of `<P>`.
- Every tag attribute needs to be enclosed in quotes.
- Every tag attribute must have a value — for example, the attribute `selected` should be `selected ="selected"` instead.

Although these standards are not a mandatory part of HTML 4.01, they are covered in this book because the XHTML standards are stricter, don't hamper HTML, and prepare you for authoring documents in other XML-based languages.

Note

Future versions of HTML are to be based on XHTML coding standards. ■

Cross-Ref

Chapter 18 provides a glimpse inside HTML5. ■

CSS 2.1

The latest CSS version is 2.1. Although version 3.0 is in development, its release might still be years away. Therefore, this book concentrates on CSS 2.1 due to its maturity. CSS version 2.0 has been around for almost a decade, is used for millions of Web pages, and is well understood by most Web designers. CSS version 2.1 combines some bug fixes, exact specifications where there was some ambiguity, and a few more properties and values. At its core, however, it is very much like version 2.0.

Although the CSS version 3.0 specification exists in draft form and has certain features adopted into certain user agents, it is far from being viable for a wide audience. As such, it is safer to stick with the existing 2.1 standard.

Cross-Ref

Chapter 38 provides a glimpse inside CSS3. ■

User Agent (Browser) Coverage

As mentioned earlier in this introduction, in 1993 Mosaic was the first widely used browser for effectively browsing the Internet. Over the years many other browsers were developed — the list is long and varied. For example, the text-only browser Lynx was developed mostly for Unix/Linux use when graphics were scarce. Other browsers such as Opera were developed to remain a pure environment, rigidly supporting the current HTML and CSS standards.

The two staples of browser-dom, IE and Firefox, continue to dominate today's market but also continue to adopt their own standards in various ways that frustrate even the most seasoned Web developer.

Over the last few years, Mac users have had Safari, a Mac-native browser. Safari hasn't been known for its speed or adherence to standards, but it does give Mac users an alternative to Microsoft Internet Explorer.

In 2009, Google's Chrome browser was released, adding yet another platform to the mix. Chrome provides many enticing features, such as a robust security framework and decent compatibility, although it is still in its infancy despite being the fourth most widely used browser. As it matures it will no doubt go through its own growing pains, including support of standard XHMTL and CSS.

So, with all these browser options, which browser(s) are specifically covered in this book? Specifically, none of the above. Rather than cover the technology of any particular browser(s), this book concentrates on the current standards of XHTML and CSS. The technologies are presented in their ratified standard form. Browser support is mentioned where appropriate, but browser-specific hacks or workarounds are not covered.

Note

Although most of the figures in this book were produced with Microsoft's Internet Explorer, it is only a matter of publishing practicality, not favoritism. ■

This decision regarding what to include keeps the book content from being too confusing while trying to cover the various quirks of various browsers, and keeps the book a manageable size.

Web 2.0

In 2004, a new World Wide Web was heralded: "Web 2.0." This new age of the Web was to facilitate interactive information sharing, interoperability, user-centered design, and collaboration.

In the next few years several outlets for this new frontier were born. They included blogs, web-based communities, hosted services, and a bevy of social-networking and collaborative sites. It seemed as though the new Web was coming into its own. Except, this new Web was nothing new.

Web 2.0 is built on the same technologies as the original and normal Web: (X)HTML, CSS, JavaScript, etc. The only difference was that the new application of the technology was much more focused on social and collaborative features. If one were to follow the evolution of the web — from academia, through business marketing, through personal use — social uses would be the next step of the evolution of the Web. This step would be a natural evolution, not the technical revolution foretold. This book takes the position that Web 2.0, as defined back in 2004, never actually took root. Instead the spirit of the use of technology on the Web reached a natural point in its evolution, using the same tools and technologies that created the Web. As such, you will not find any specific Web 2.0 coverage within this book, but will be able to employ the building blocks that are covered for a wide range of purposes, including social and collaborative online tools.

Terminology

To stay progressive with the evolution of the Web and its direction today, this book uses less technical and more progressive terminology.

For example, you will seldom, if ever, see the words "page" or "Web page" used to refer to Web content in this book. That's because as the Web has matured as a publishing medium, words such as "document" are much more apt for describing content on the Web.

Similarly, the word "browser" is a bit passé, and is therefore rarely used. In the past, applications such as Mosaic, Mozilla, Firefox, Opera, and Internet Explorer were the only game in town when it came to accessing the Web. Such applications, which were primarily used to "browse" content on the Web, were aptly dubbed "browsers."

However, the devices and applications used to access Web content today are much broader:

- Personal electronic devices
- Onboard vehicle systems
- Entertainment system controllers
- Mall kiosks

Many of these Web-enabled applications are not like traditional browsers. They may access data differently, present data differently, and might be controlled differently than a browser. A better term for these applications is user agent, which basically means "something that enables a user to access data," which is what each of these does. For that reason, get used to seeing *user agent* instead of *browser*.

Who Should Read This Book?

This book is geared toward a wide audience. Readers who are just getting started with HTML and Web content will benefit the most, as this book provides both a solid learning foundation as well as ample reference material for later perusal. Experienced users will find the chapters covering new standards and technologies to be the most useful, but also will appreciate having a comprehensive reference for consultation.

Although the Web is technical in nature, this book boils down the technology into simple and straightforward terms. Whether you qualify as a computer scientist or as a computer neophyte, you will be able to understand, adopt, and deploy the information throughout this book.

This Is Not a Web Design Book

This book teaches the basics of HTML elements, how to integrate said elements, and finally how to layer CSS over the top. Design books generally skimp on the building-block detail, only covering how to best use the elements to achieve cosmetically pleasing results. While each type of book does cover principles of the other, the cross-over content is not comprehensive.

Typically, both approaches do not appear in the same book due to size constraints. The other reason why the two approaches are different has to do with the separation of content and design. This book concentrates on the content portion of Web design, whereas other design-centric books cover the design (visible attributes).

Tip

Wiley publishes many Web design books that can be paired with this book to provide a wide range of skills and techniques for creating technically correct and visually pleasing documents.

Two such recommendations include:

- ***Creating Web Sites Bible*, Third Edition, by Philip Crowder and David A. Crowder (2008).**
- ***Beginning CSS Cascading Style Sheets for Web Design*, 2nd Edition, by Richard York (Wrox, 2007).**

Visit the Wiley website (`www.wiley.com`) and search on "web design" to find other books applicable to your needs. ■

What Is Contained in This Book?

This book is divided into four major sections, plus five appendixes.

Part I: Creating Content with HTML

This part of the book covers the basics of HTML — the tags, attributes, and structure that make up the language. You learn how to structure a document, format text, and incorporate multimedia. You also learn basic and advanced scripting to lend a dynamic edge to your documents.

Part II: HTML Tools and Variants

This part of the book covers utilities to help you author, validate, and troubleshoot your documents. A few useful HTML variants and extensions — including XML and XHTML Basic — are also covered.

Part III: Controlling Presentation with CSS

This part of the book covers the basics of CSS, the syntax of CSS selectors, valid properties and values, and how to use CSS properties to effectively format the various portions of your document. You will also learn how to format a document for printing using CSS media types.

Part IV: Additional CSS Tools

The last part of this book covers additional CSS topics, including advanced layout, user interface styles, testing and validating CSS, and some CSS tips and tricks.

Reference Appendixes

The appendixes provide a quick reference to the material covered in detail throughout the rest of the book.

Tip

See the Table of Contents for a breakdown of chapter topics in each part. ■

How to Use This Book

This book can be used in a variety of ways depending upon your skill level and intent.

The sequential read

If you need to learn HTML and CSS from beginning to end, then a sequential read — reading the chapters in order from beginning to end — is for you. The chapters are designed to introduce topics in a particular order to get you started and build toward more advanced topics.

For a tutorial approach, choose a sequential read.

A targeted or random read

If you need only a refresher of certain material or want to learn in a different order than the chapters provide, then a targeted or random read — finding a topic in the table of contents or index to read, or reading chapters in a different order than numbered — is for you. Although the chapters were written to build on one another, they also are topical and encapsulate individual subjects. Find a chapter with information you need to learn and read it, or find a section within a chapter and read it alone.

For a referential approach, choose a targeted or random read.

Conventions and Features

Many different organizational and typographical features are used throughout this book to help you get the most from the information contained within.

Tips, Notes, and Cross-References

Whenever the author wants to bring something important to your attention, the information appears in a Tip, Note, or Cross-Reference. These elements are formatted as follows:

Tip

This information is important and is set off in a separate paragraph with a distinct look. ■

Tips generally are used to provide information that can make your work easier — special shortcuts or methods for doing something more easily than the norm.

Notes provide additional, ancillary information that is helpful but somewhat outside the scope of the material presented.

Cross-references indicate other places in the book you'll find information pertinent to the topic at hand.

Code

It is often necessary to display code (HTML tags, JavaScript commands, script listings) within the text. This book uses two distinct conventions, depending on where the code appears.

Code in text

A special font is used to indicate code within normal text. For example:
`<body id="COMPONENT-body-0001" onLoad = "displaygraphics();">`.

Code listings

This code is set apart and indented from regular text, as follows:

```
 Code listings appear in specially formatted listings, in a different
font, similar to these lines.***
```

Companion Website

A companion website has been created to help support this book. It contains code from the book and examples within, as well as extra material not contained in this book. The website can be found at `www.wiley.com/go/htmlbible5e`.

Part I

Creating Content with HTML

IN THIS PART

CHAPTER 1

What Is a Markup Language?

The World Wide Web is a technology beast. If you have read this book's introduction, you should have at least a passing familiarity with how the Web started — its humble beginnings to bring cross-referenced textual documents to the masses via the connectivity of the Internet.

You are reading this book, so it's a good assumption that you are familiar with what the Web has become today — a collection of technologies capable of transporting numerous media across the Internet for consumption directly on your desktop.

However, it's important not to forget the Web's humble beginnings because the technologies used for the very first simple documents are still in use today, and must be understood. This chapter helps frame the reasons why.

IN THIS CHAPTER

What Are We Doing Here?

Why are we diving into technical topics instead of talking about how to create Web documents? Well, technically we are talking about how to create Web documents. The more you know about the technology behind the Web, the better prepared you will be to use the technology to your benefit, and the easier it will be to create Web documents.

Note

If you really do want to just dive into creating documents, check out Chapter 19, "Web Development Software," which covers tools you can use to quickly create documents without knowing the underlying technology behind it all. However, keep in mind that such tools do not always accomplish the goal you desire and sometimes their results need manual tweaking — tweaking that you will learn to perform throughout the other chapters in this book. ■

So back to the question: What are we doing here?

Answer: Web documents are created using several different technologies. The main technology is Hypertext Markup Language (HTML). HTML is responsible for telling a Web browser (e.g., Microsoft Internet Explorer, Mozilla Firefox, Opera, Mac Safari, Google Chrome, and so on) how text and other objects in a Web document should appear. Whether the text should be small, large, bold, underlined, or right or left justified is largely determined by the HTML embedded in a Web page.

As a consumer of Web pages, you rarely experience HTML directly; it's hidden from the end user by the browser. However, as a creator of content, you need to be intimately familiar with HTML and its uses, which is why we are starting from scratch and covering some basics first. Don't worry, the good stuff is right around the corner and we will get started creating actual content soon enough.

Understanding Hypertext

By its very nature, the Web and its content overcome many of the limitations of standard, linear text. This concept is best illustrated by a comparison of a book (in particular, a reference book) to the Web. For example, consider a cross-reference in a book. Accessing the cross-reference requires *you* to look up the page number, textual reference, or other object being referred to. On the Web, the reference is (usually) a single mouse click away.

Also, documents on the Web can be designed to vary depending on the user accessing them. Books, conversely, remain static objects no matter who is reading them.

The word "Hypertext" was created along with other Internet terms and technologies during the evolution of the Web. It was coined to describe documents that could change, redirect, and otherwise overcome the linearity of normal text. In short, "Hypertext" describes text on the World Wide Web.

Understanding Markup Instructions

Markup languages are not a difficult concept to grasp; most of you have "marked something up" at one point or another. For example, suppose you wanted someone to highlight a paragraph in this book. It would be fairly easy for you to instruct that person to do what you wanted — you could simply hand the person a highlighting pen, point to the paragraph, and ask the person to highlight it.

Note

Highlighting is only an example of what you might want to happen to a piece of text. You might want some text to be larger, bolder, underlined, or otherwise changed. Highlighting is used in this chapter as a simple, real-world example. ■

Consider the paragraph shown in Figure 1-1, highlighted in Figure 1-2.

FIGURE 1-1

A simple paragraph

Welcome to On Target Games, the online home of the best-selling game, Vanguard Odyssey. Enjoy browsing the site and don't forget to check out the updates section.

FIGURE 1-2

The same paragraph, highlighted

Welcome to On Target Games, the online home of the best-selling game, Vanguard Odyssey. Enjoy browsing the site and don't forget to check out the updates section.

This is a relatively easy task to ask of someone and have executed, because you, and most other people, understand the concept of paragraphs. You point to a paragraph and the person doing the highlighting knows the boundaries — the beginning and the end of the text to be highlighted. If the individual were really dense or needed more explicit instructions, you could write the instructions on or near the paragraph, as shown in Figure 1-3.

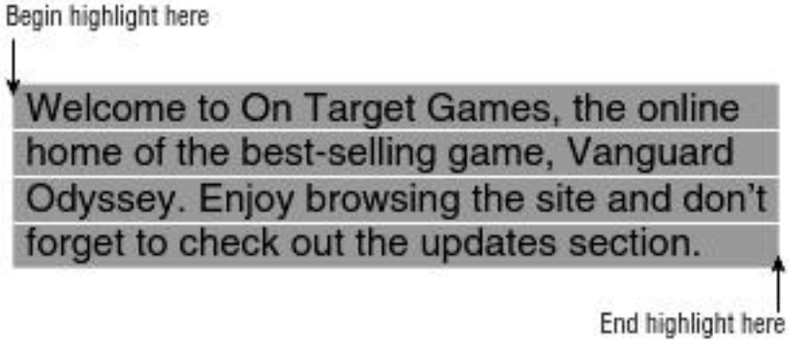
Note

Writing explicit editing instructions in or around text is generally known as *marking up* text. ■

Notice how the instructions "bookend" the portion you want affected. In other words, the "begin" instruction appears before the text to be highlighted, while the "end" instruction appears afterward. This is an important concept in text markup.

FIGURE 1-3

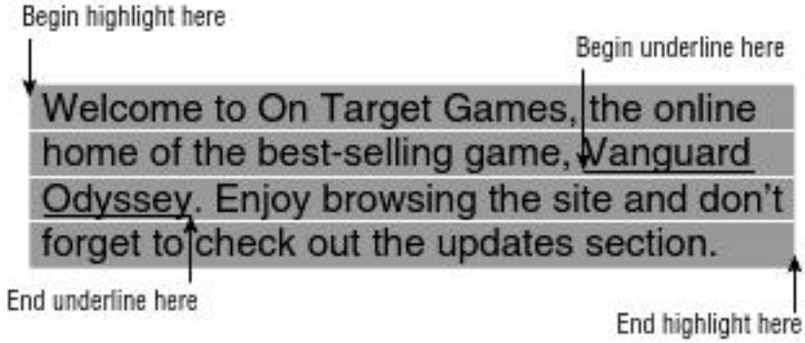
Explicitly designating the area to be highlighted by marking up the paragraph



You might want more formatting to be done to the text. For example, suppose you wanted "Vanguard Odyssey" underlined. Specifying that additional formatting could resemble the paragraph shown in Figure 1-4.

FIGURE 1-4

Multiple formatting instructions might appear close to one another, or even nested within one another.



Understanding Markup Language

On the Web, you aren't dealing with humans; you are dealing with computers and software — namely, Web browsers. You create content specifying how the browser should display it (highlighting certain pieces of text, and so on). When the browser displays the page, it applies the appropriate formatting accordingly so the user sees the text and document as you intended. You need a way to mark up the text so the browser understands it.

In the early 1990s, a new programming-like language was created, Hypertext Markup Language, or HTML. Don't let the word programming scare you — it is used here to put the word "language" in context; it's not really programming, as you will see. The language was created to provide a way for users to mark up documents so Web browsers could display certain elements of the document in italics, underlined, and so on.

Several requirements must be considered when telling a computer how to format text. A short list of the requirements includes the following:

- The instructions should follow a stringent set of guidelines.
- The instructions should be included in the textual document.
- The instructions should be invisible to the end user.

- The instructions should tell the display device (usually a Web browser) where to start and end, and how to apply the formatting specified.

Note

The first item in the preceding list, requiring a "stringent set of guidelines," is very important. As with most programming languages, a strict set of guidelines and syntax is necessary to ensure that the programmers (Web designers) create programs (Web pages) that the computers (Web browsers) can understand. Throughout this book, I will continually stress the standards created by organizations such as the World Wide Web Consortium (W3C), the folks behind the World Wide Web and its related standards. ■

Essentially, a *markup language* is a systematized and standardized markup instruction set.

Consider how such a language would work. As in the earlier example about highlighting, instructions could be appended to the paragraph similar to that shown in Figure 1-3. However, because the Web page needs to be in electronic text form only (no handwriting allowed!), the document would end up resembling something like this:

```
Begin Highlight Here Welcome to On Target Games, the online home of
the best-selling game, Begin Underline Here Vanguard Odyssey End
Underline Here. Enjoy browsing the site and don't forget to check out
the updates section.  End Highlight Here
```

It's difficult to tell where the text and markup begin and end when the markup is used in this way. It would be much better if the markup instructions were delimited by something so that you, and the Web browser, could tell where and what they were.

Thankfully, in HTML the markup instructions are indeed delimited. They are enclosed in angle brackets — more commonly known as "less than" and "greater than" signs (< and >). Furthermore, the directives don't need the words "begin" or "end." The beginning marks simply contain a keyword corresponding to what the markup should accomplish, and the ending marks include a slash (/). For example, the underlining markup directive is simply "u" (for underline) and it appears as shown in the following text:

```
Welcome to On Target Games, the online home of the best-selling game,
<u>Vanguard Odyssey</u>. Enjoy browsing the site and don't for-
get to check out the updates section.
```

The <u> designates the beginning of the underline and the </u> designates the end. This paragraph rendered in a Web browser would resemble what is shown in Figure 1-5.

Similarly, in HTML, bold is represented by "b" (and), italic by "i" (and), and so on. Other markup instructions and directives have similar tags. These tags are inserted into Web pages, and the Web browser reads the page and uses the tags to properly format the text and other items on the page.

FIGURE 1-5

The paragraph in a Web browser

Summary

What does all this mean? There are some basic technologies underneath the surface of the Web to which you must pay attention. HTML is the backbone of these technologies, and knowing it is the key to successful Web design. Understanding markup concepts is key to understanding proper HTML use.

CHAPTER

2

HTML Values and Units

In the previous chapter you learned what markup language is and how it relates to HTML and the Web. Expanding on these basics, you can add attributes to your HTML tags to further control their effect on your documents.

IN THIS CHAPTER

Basic Tag Attribute Format

Most HTML tags support one or more attributes. These attributes are included in the opening tag using a standard format, as follows:

```
attribute_name="attribute_value"
```

For example, the `border` attribute is used with the `<table>` tag to control the width of the border in and around a table in the document. The border attribute resembles the following when actually included in the `<table>` tag:

```
<table border="1">
```

Pay close attention to the following rules regarding attributes:

- Any attributes in an HTML tag need to appear after the HTML tag name.
- The attribute name must be followed immediately by an equal sign (=).
- The attribute value needs to come immediately after the equal sign.
- The attribute value must always be enclosed in quotes, either single or double.

Note

In previous versions of HTML, some attributes — namely, those with default values — did not need to have a value associated with them. However, in HTML all attributes must have a value included inside the tag with their declaration. In short, you should always provide a value with tag attributes. ■

Several different types of values can be used as values for attributes:

- Text (single words, no spaces)
- Numbers (unsigned)
- Color values (color names or color values)

In the case of color values, several options can be used to specify a particular color:

- Color names (blue, black, red, and so on)
- Color values (in hexadecimal)
- Color values (in decimal)

The color name method is very straightforward; you simply specify a color as the value of the attribute. For example, in the following color attribute snippet, the color is set simply to `"blue"`:

```
color="blue"
```

This method accepts only a preset number of colors defined by HTML — approximately 147 different colors that can be found listed on sites such as `www.w3schools.com/html/html_colornames.asp`.

The hexadecimal and decimal methods of specifying colors are slightly more complex because they allow you to actually mix colors by specifying custom amounts of the primary colors: red, green, and blue. The correct hexadecimal format follows:

```
"#RRGGBB"
```

The color specification must begin with a pound sign (#) and be followed by six digits — the first two digits corresponding to the value of red, the second green, and the third blue. Again, keep in mind that these values are hexadecimal, *not* decimal. Consider the following codes and corresponding values:

```
#FF0000          Red
#00FF00          Green
#0000FF          Blue
#FF00FF          Purple (Red and Blue)
#000000          Black
#FFFFFF          White
```

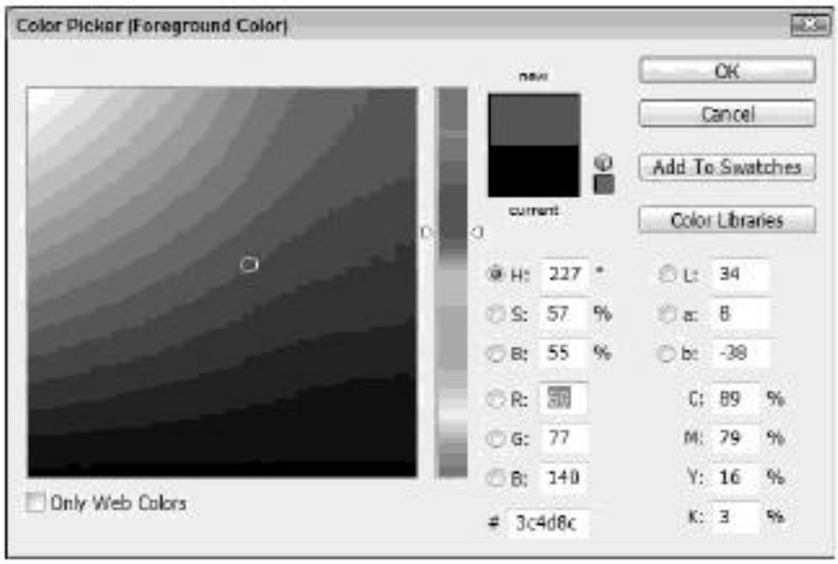
For example, to set a color attribute to purple, you could use the following code:

```
color="#FF00FF"
```

This allows more control over the actual color, but requires you to compute the value of the color in hexadecimal. Thankfully, most graphic editing programs contain features to display or convert color values in hexadecimal format. For example, Figure 2-1 shows the color selection dialog in Adobe Photoshop, which includes a hexadecimal value of the current color.

FIGURE 2-1

Most graphic editing programs, like Adobe Photoshop shown here, include methods to specify colors in both decimal and hexadecimal values.



The other format supported by the color attribute is the color's RGB (red, green, blue) value in decimal format. Instead of being prefixed by a pound sign, the RGB decimal format uses the following format:

```
color="rgb(R,G,B)"
```

In this case, the values of the colors are specified as values between 0 and 255, or percentages (values between 0% and 100%). For example, to set the color to purple (max red, no green, max blue), you would use either of the following codes:

```
color="rgb(255,0,255)"
color="rgb(100,0,100)"
```

Throughout this book, the applicable attributes are discussed along with the tags to which they apply.

Note

See the next section, "Common Attributes," for attributes common to most tags. ■

Common Attributes

Several attributes are available and applicable to most tags in HTML. These attributes serve the same general purpose regardless of the tag with which they are used. The following sections describe these tags and the purpose they serve when you apply them.

Tag identifiers – IDs and classes

As you will learn in the style and scripting sections of this book, sometimes it is advantageous to identify particular tags so you can refer to them by other methods in the document.

IDs

The `id` attribute effectively assigns a unique identifier to a tag. For example, if you use a `<table>` to contain inventory data, you might use the `id` attribute to name the table `inventory`:

```
<table id="inventory">
 ...
</table>
```

Note

When using the `id` attribute, keep in mind that each tag should have a unique value for its `id` attribute. ■

Locally — that is, within the tag — the `id` attribute has no real effect. However, scripts can access and manipulate tags based on their `id` attribute.

Cross-Ref

For more information on how scripts can access tags based on their `id` attribute, see Chapters 16 and 17. ■

As you design your pages, consider whether you will need to reference any of your tags by outside means (scripts and so on).

Classes

Classes are similar to IDs in that they help identify tags in the document for use by other methods. However, unlike IDs, which should be unique, classes can (and should) be applied across several tags in your document.

Applying classes to tags is similar to applying IDs and other attributes. For example, to apply a class `"emphasis"` to a `table` tag, you would use code similar to the following:

```
<table class="emphasis">
```

As with the `id` attribute, the `class` attribute doesn't directly affect the tag to which it is added. What the `class` attribute does do is link the tag to CSS styles that also reference that specific