UNIT
B
HTML5 and CSS3

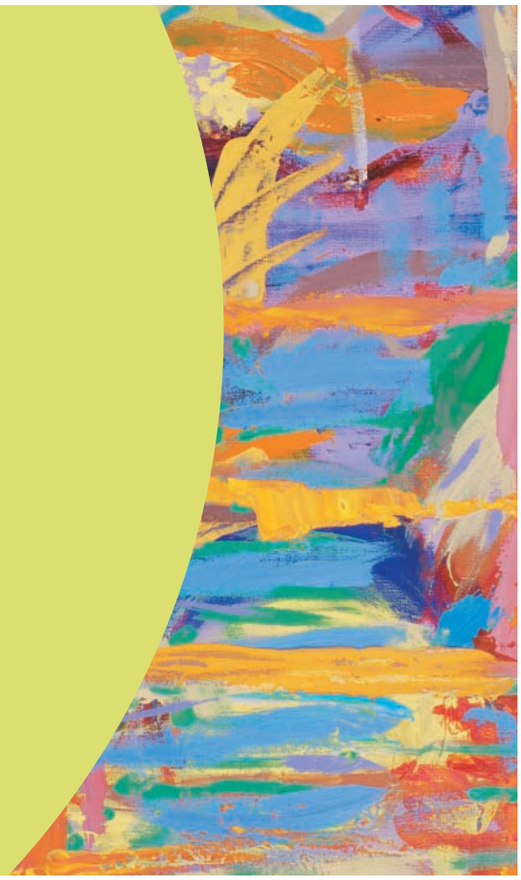
Getting Started with HTML

Files You Will Need:
No files needed.

After you plan out a Web site, you can begin to create it with nothing more than a basic text editor. Web pages are written in **Hypertext Markup Language (HTML)**, which is a standardized format for specifying the structure of a Web page. You can add HTML code to the text you want to be part of your Web pages in order to indicate the different types of content present. To test your pages and preview their appearance, you can open them in a Web browser or other user agent that interprets your HTML code and represents your Web page contents in a consistent way. Faduma Egal, the art director at Great Northern Web Solutions, has approved the plan you created for the Lakeland Reeds Bed & Breakfast Web site. She wants you to start work on the site by creating a home page containing basic information about the bed and breakfast, including contact information. In this unit, you'll learn more about HTML, and then create and test the main Lakeland Reeds B&B Web page.

OBJECTIVES

- Assess the history of HTML
- Compare HTML and XHTML
- Create an HTML document
- Set up the document head and body
- Add Web page text
- Preview your Web page
- Implement one-sided tags
- Validate your HTML code



UNIT
B
HTML5 and CSS3

Assessing the History of HTML

HTML was created in 1991 by Tim Berners-Lee. In the succeeding years, the language has undergone several major revisions, along with a couple periods of reinvention when the underlying purpose and direction of the language were reevaluated. Along the way, a few major themes emerged, all of which are important factors in understanding HTML today.  You start your work on the Web site by exploring the history of HTML and the factors that have influenced the current version of the language, HTML5.

DETAILS

A few factors have heavily influenced the development of HTML:

- **Process and evolution**

  The first two versions of HTML were defined and published by the **Internet Engineering Task Force (IETF)**. In 1994, the **World Wide Web Consortium (W3C)** was founded to take on the responsibility of maintaining the language's standards, which it continues to do today. Although HTML has gone through several major versions (see Table B-1), in practice the language is constantly under transformation. Through ongoing proposals and debates, the W3C facilitates the process of clarifying and amending the existing specification to incorporate new features while keeping the language consistent. The W3C also maintains a vision for the next steps in the evolution of HTML in an attempt to clarify what developers and software companies want prioritized in future versions of the language.

- **Semantics vs. presentation**

  The publication of HTML 4 in 1997 marked a major turning point in the development of the language. Companies that created Web browser software had begun to anticipate the W3C standards by adding their own extensions to HTML that generally worked only on a particular brand of browser. Between these extensions and a mushrooming of overlapping features codified in previous versions of HTML, it was becoming clear that the language needed to narrow its focus. HTML 4 codified HTML as a **semantic** language, meaning its intended use was to indicate the meanings of elements such as headings and paragraphs in a Web page, but not to tell Web browsers how the elements should appear. The new version of HTML coincided with the rollout of a companion language, **Cascading Style Sheets (CSS)**. CSS is a presentational language, meaning that it's designed for describing the appearance of items.

  Many features of previous HTML versions were presentational rather than semantic and, thus, didn't fit the new model of HTML. However, because so many Web pages had already been written using these features, and existing Web browsers and other user agents supported them, the W3C designated these features as **deprecated**, meaning that, while these features could still be used, their use was no longer recommended, and alternatives to their use were available. This step gave Web designers time to learn new ways of coding and update their Web pages gradually while maintaining support for the current state of the Web.

- **Web application support**

  In the years since the HTML 4.01 specification was finalized, common uses of the Web have changed and grown in significant ways. The core updates that are part of HTML5 include integrated support for new features, such as embedding video, as well as enhancement of longstanding features, such as Web page forms.

TABLE B-1: The evolution of HTML

version	year	major changes
HTML 2.0 & 3.2	1995–97	Codified the current state of HTML
HTML 4 & 4.01	1997–99	Added support for CSS and several newly developed HTML features
HTML5	2008–present	Enriches semantic options, along with new features and enhancements

HTML and Web browser versions

In the early years of the Web, browser creators Microsoft and Netscape added to their browsers proprietary features that weren't supported by their competitors. This provided opportunities for Web developers to add new elements to their Web pages. However, it also meant that any Web page incorporating these features would be displayed differently depending on which browser was used to open it. This situation created difficulties for developers trying to reach the widest possible audience with their Web pages. Eventually, the major browser companies recognized that creating their own features would not be a major key to gaining or maintaining market share, and they have joined with developers to support a largely standards-based Web.

On the flip side, recent updates to Web standards have not always been quickly or fully adopted by makers of browsers and other user agents. As a result, when considering whether to use a particular HTML feature in your Web pages, it's important to research which user agents support it and how widely the supporting user agents are in use. In some cases, implementing new features will need to wait until your target audience has caught up with available Web technologies.

UNIT
B
HTML5 and CSS3

Comparing HTML and XHTML

Efforts to enhance Web standards didn't stop with the publication of the HTML 4.01 specification in 2002. Instead, work began to make HTML interoperable with another markup language known as **Extensible Markup Language (XML)**. The result was a new specification known as **Extensible Hypertext Markup Language (XHTML)**, which for several years was seen as the successor to HTML. Today work continues on both HTML and XHTML, and each has similar, distinct applications.  As you prepare to design Web pages, it's important to understand the differences between HTML and XHTML, and when each might be appropriate in your work.

DETAILS

HTML and XHTML both have distinct advantages:

- **XHTML is XML-compliant**

  In addition to HTML, many other markup languages exist for specialized applications. Many of these languages are subsets of XML. Like HTML, XML enables users to describe the structure of a document. However, because XML is a more generic language than HTML, it enables users to describe any kind of document, instead of only Web pages. XHTML grew out of work in the late '90s to make HTML comply with the rules of XML in order to facilitate the interoperation of Web pages with documents coded using other XML-based languages. The first phase of this work resulted in the XHTML 1.1 standard.

  Adapting existing HTML standards to adhere to XML syntax required that Web page authors change a few aspects of the way they write code, which are outlined in Table B-2. You'll see examples of these aspects as you begin to create Web pages.

- **HTML can handle coding errors and quirks**

  As HTML standards changed and grew during the '90s, flexibility became an important part of the language. Even if Web page authors made minor mistakes in writing code, user agents could often interpret the code loosely, rather than hewing unerringly to rules, and successfully display their best interpretations of what the authors intended. XML, however, does not tolerate errors. Thus, if a user agent encounters a coding error while attempting to display a Web page coded in XHTML, the user agent must display an error message rather than read the code more loosely.

  While work initially went forward to create XHTML 2.0 as a successor to XHTML 1.1, some members of the Web standards community dissented. The W3C's focus solely on XHTML as the path forward meant that Web pages containing coding errors would not be displayed correctly or at all. It also meant that any new features under consideration for the next version of XHTML would be unavailable in the billions of existing Web pages written in HTML unless their code was edited to comply with the rules of XHTML. Some community members saw this as an unnecessary barrier and instead began working together to build on the HTML 4.01 standard in order to create a specification that integrated markup features geared toward modern Web usage.

- **Moving forward: coexistence**

  Today, the W3C is improving and expanding both XHTML and HTML, seeing the two as parallel languages with their own reasons to exist. In this book, you'll learn how to create Web pages in both HTML and XHTML, so you can make your own decisions about which language is most appropriate for projects you work on in the future.

TABLE B-2: Differences between HTML and XHTML

aspect	HTML	XHTML
Tag nesting	Tags may be closed out of order	Tags must be closed in the order opened
Tag case	Tags may be written in upper or lower case	Tags must be written in lower case
Tag closure	Closing tags may be omitted for some elements	All opening tags must be matched with closing tags
Attribute-value pairs	Value may occur without attribute name	Attribute-value pair required
Empty elements	Empty elements do not need to be closed	Empty elements must be closed
Script and Style elements	Elements may be placed within HTML document	Elements must be in separate documents and linked to XHTML document
Attribute values	Values may be enclosed within quotes	Values must be enclosed within quotes
ID and Name attributes	Either attribute may be used	ID attribute may be used, but Name attribute prohibited
Character codes for special characters	Codes recommended	Codes required

The W3C and the WHATWG

After XHTML 1.1 was finalized, the W3C moved forward with adding new features to XHTML, and drafted a proposal for XHTML 2.0. However, a number of community members, including several major technology companies, felt that HTML, rather than XHTML, would better serve the future of the Web, and proposed that the W3C change course. When a W3C committee voted against this proposal, these companies formed the **Web Hypertext Application Technology Working Group (WHATWG)** to begin a process of creating a new HTML specification. Over time, the two organizations bridged their differences and agreed to collaborate in creating HTML5, with the W3C coordinating work on XHTML5.

Before finalizing the latest version of HTML, the two organizations are following a process of soliciting input and incorporating feedback from members of the Web community, including makers of user agents and other software and Web developers. As a result, the development process will likely continue for several years before all parties involved finalize a single standard. In the interim, however, different features of the developing specification may become available at different times.

UNIT
B
HTML5 and CSS3

Creating an HTML Document

An HTML document consists solely of text. As a result, you can create a Web page in a text editor such as Notepad, which is included with all current versions of Windows or TextEdit, which is part of Mac OS X. To create a Web page, you enter text that you want to display on the page along with HTML codes known as **tags**, which specify how a user agent should treat each item in the document. While most tags occur in pairs, some tags, known as **one-sided tags**, are used by themselves.  You decide to use a text editor to create the home page for the Lakeland Reeds Bed and Breakfast.

STEPS

1. **Start your text editor**
   A new, blank document opens, as shown in Figure B-1.

QUICK TIP
To read more about any HTML tag used in these steps, see Appendix A.

2. **Type <!DOCTYPE html>, then press [Enter]**
   This one-sided tag creates an element known as the **DOCTYPE declaration**, which lets user agents know that the document contents are written in HTML.

3. **Type <html>, press [Enter] twice, then type </html>**
   A tag pair assigns meaning to a Web page **element**, which is a specific component of the page, such as a paragraph or a heading. You place the **opening tag** at the start of the element you are marking and the **closing tag** at the end. HTML tags always start with an opening angle bracket (<) and end with a closing angle bracket (>). A closing tag is the same as its corresponding opening tag except that the opening angle bracket is followed by a slash (/). The text between the angle brackets specifies the HTML element type being applied to the selection. The html tag pair marks the beginning and the end of the Web page.
   Your document should match the one shown in Figure B-2.

4. **If you are using TextEdit on a Mac, click Format, then click Make Plain Text**

5. **Click File, then click Save**
   The Save As dialog box opens.

6. **Navigate to the drive and folder where you store your Data Files, open the Unit B folder, then open the Unit folder**

TROUBLE
If you are using TextEdit on a Mac, uncheck the Hide extension check box if necessary.

7. **In the File name box (Windows) or Save As box (Mac), type index.html**
   The standard name for the main page of a Web site is "index." The .htm or .html extension signifies that a file is written in HTML.

8. **If you are using Notepad in Windows, click the Save as type list arrow, then click All Files (*.*)**

9. **Click Save**
   The index.html file is saved to your storage location.

Other Web page creation software

Many other programs are available that allow you to create Web pages visually by clicking buttons and using drag-and-drop to place items on a page. However, creating your first Web pages by entering HTML directly—sometimes referred to as **hand-coding**—is one of the best ways to get familiar with HTML and the underlying structure of a Web page.

FIGURE B-1: A blank document in Notepad

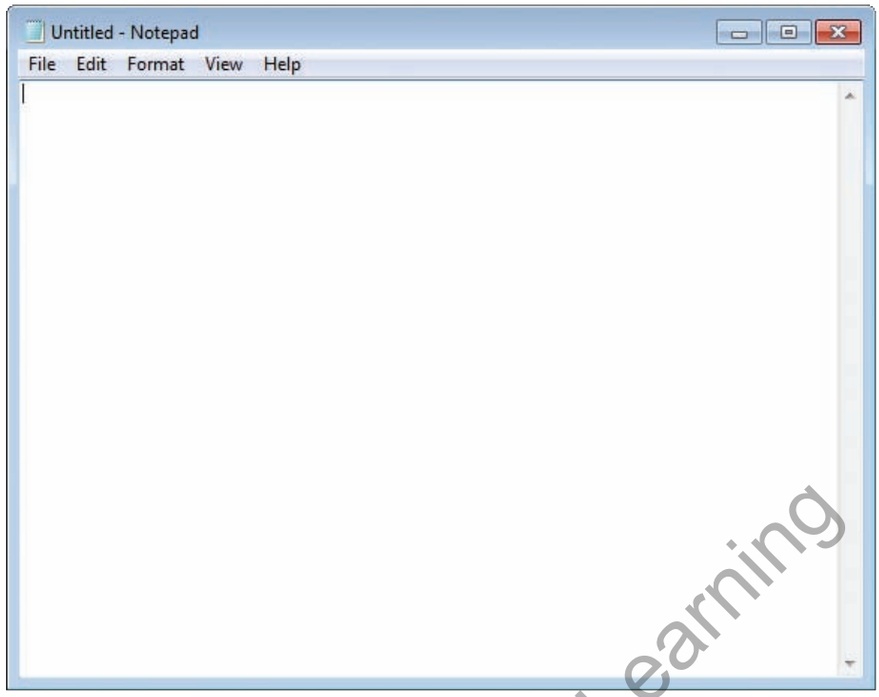
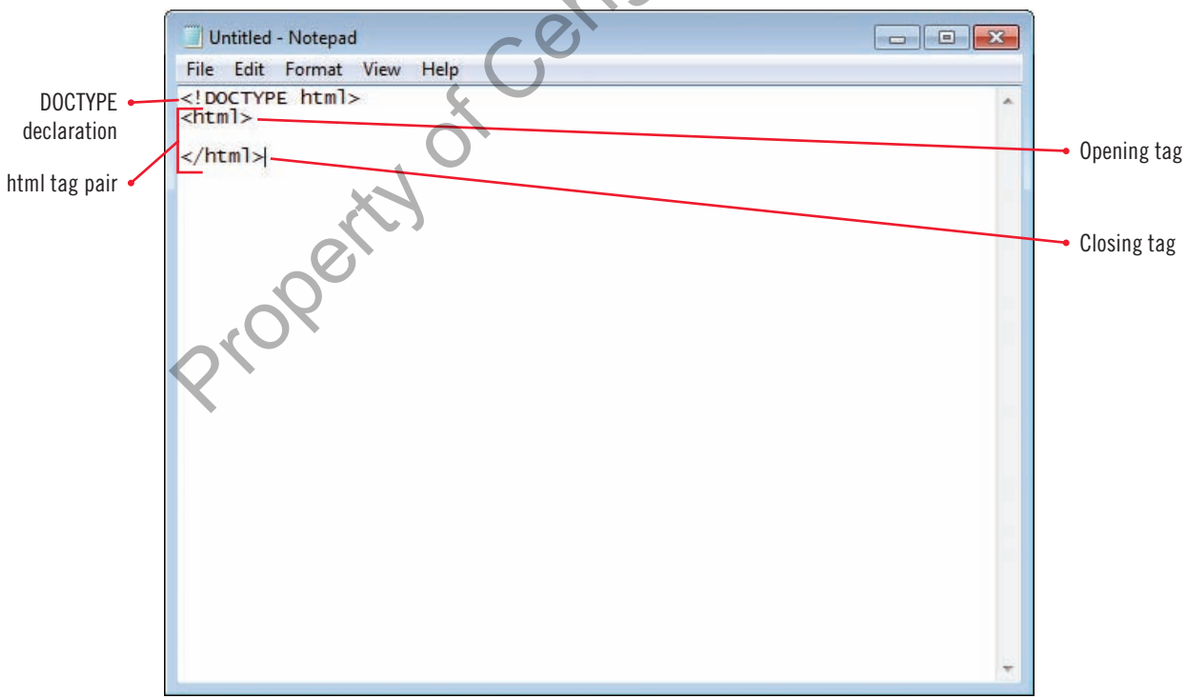


FIGURE B-2: The basic structure of the Web page



UNIT
B
HTML5 and CSS3

Setting Up the Document Head and Body

Within the html tag pair for a Web page, the document contents are divided into two sections. The head section contains elements that are not part of the main Web page, such as the page title that appears in a browser's title bar. The contents of the **body section** are visible in the main window of a Web browser and include elements like paragraphs and headings. Both the head and body tag pairs are located within the html tag pair. This situation is known as **nesting**, and most elements in the code for a Web page are nested within one or more other elements.  As you continue creating the structure for the Lakeland Reeds Web page, you add the head and body sections to the page.

STEPS

1. **Click in the blank line between the opening and closing html tags, press [Spacebar] twice, then type <head>**
   Adding two spaces before a nested tag makes it appear indented. As your Web page code becomes longer and more complex, these indentations make it easier to identify the beginning and end of an element at a glance. Figure B-3 shows HTML code with several layers of nested tags that are formatted with indentations.
2. **Press [Enter] twice, press [Spacebar] twice, then type </head>**
3. **Press [Enter], press [Spacebar] twice, then type <body>**
4. **Press [Enter] twice, press [Spacebar] twice, then type </body>**
5. **Click in the blank line between the opening and closing head tags**
6. **Press [Spacebar] four times, type <meta charset="utf-8" />, then press [Enter]**
   The meta element enables you to pass information about a Web page to user agents that open it. The text following the name of the tag in the code you typed is an **attribute**, which you can use to provide additional information about an element. The charset attribute specifies the **character encoding**, which is the system user agents should employ to translate the electronic information representing the page into human-recognizable symbols, such as letters and numbers. Figure B-4 shows the completed Web page structure.
7. **Save your work**

QUICK TIP
To save your work without using the menus, you can press **[Ctrl]+[S]** (Windows) or **[command]+[S]** (Mac).

Describing nested elements

An element nested within another element is called a **child element** of the enclosing element, and the enclosing element is known as the **parent element**. Two elements that are both children of the same element are known as **sibling elements**. In the code

```
<html>
  <head>
    <meta charset="utf-8" />
```

the head element is both a child of the html element and the parent of the meta element. In addition, here the html element is the **grandparent element** of the meta element, which can be referred to as a **grandchild element** of the html element.

FIGURE B-3: HTML code containing multiple layers of nested elements

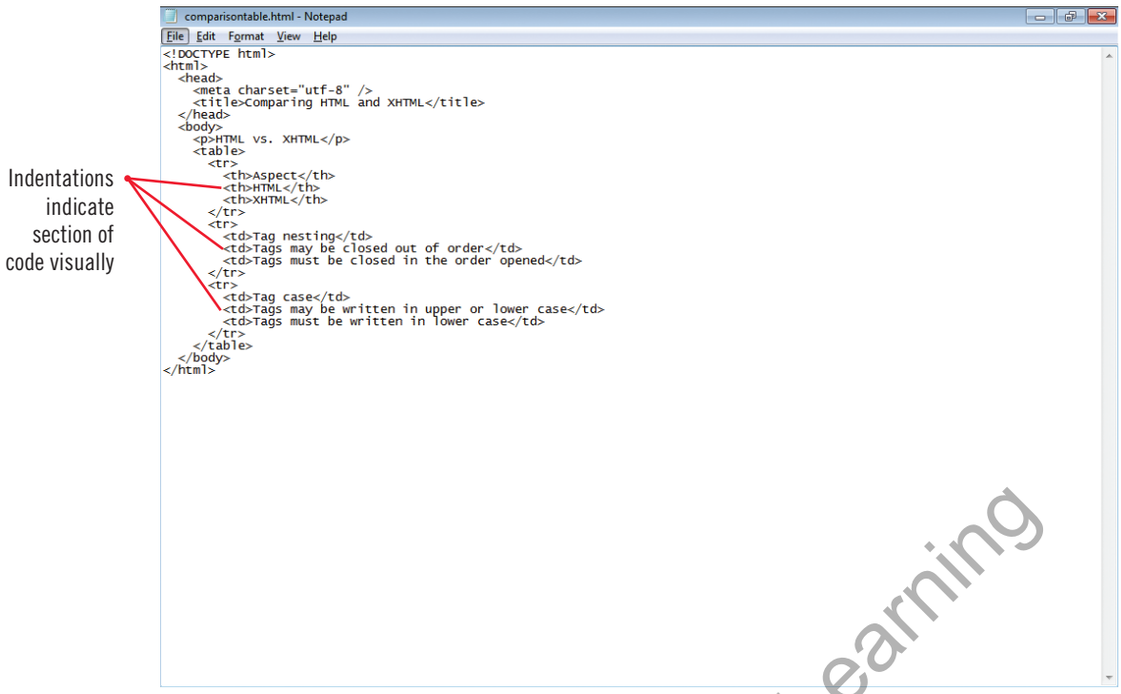
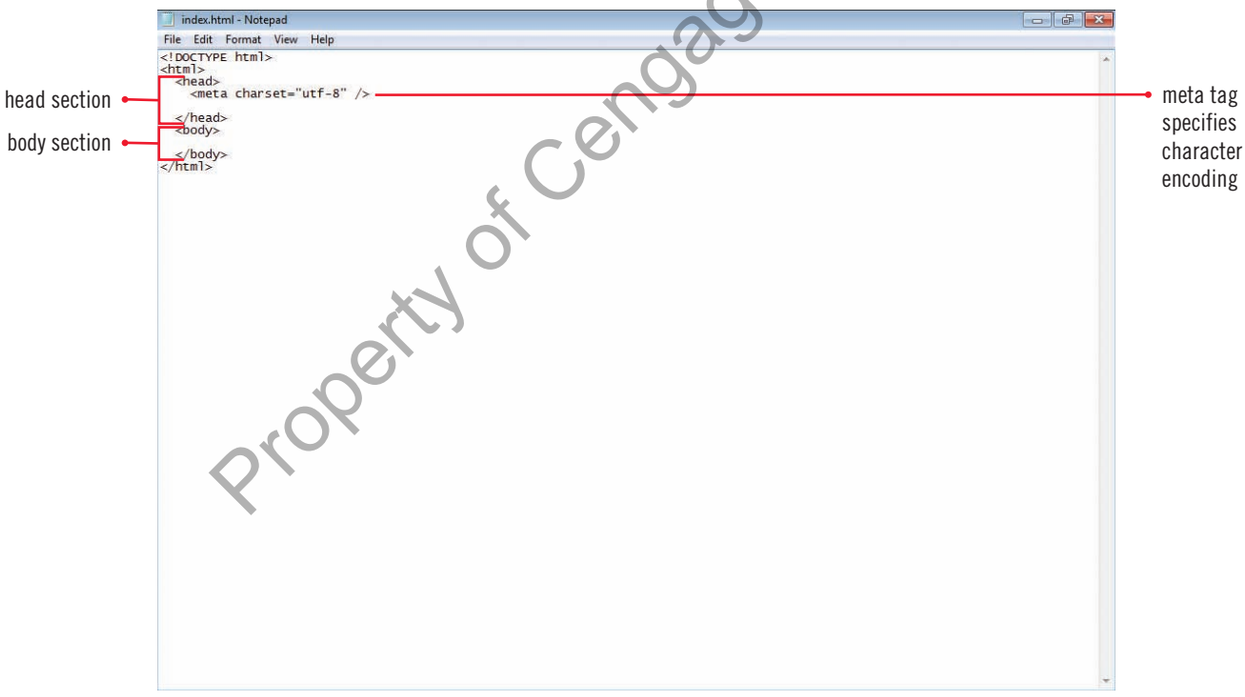


FIGURE B-4: Completed Web page structure



HTML attributes

Many, but not all, HTML elements allow you to set attributes to specify details about a given element's properties. To use an attribute, you provide two pieces of information: an attribute name and the value you are assigning to the attribute. Together, these two pieces are known as an **attribute-value pair**. In the element <meta charset="utf-8" />, *charset* is the attribute name, and *utf-8* is the attribute value. In a tag pair, you specify any attributes in the opening tag, never in the closing one.

UNIT
B
HTML5 and CSS3

Adding Web Page Text

Because an HTML document is simply a plain text document that includes HTML codes, entering text for your Web pages is as simple as typing it. You then add the appropriate HTML tags to specify the element type for each text item on the page.  Figure B-5 shows a sketch you created of a simplified initial version of the Lakeland Reeds Web page. You decide to enter the page title that will appear in the title bar of viewers' browsers, along with the basic information about Lakeland Reeds.

STEPS

QUICK TIP
This line is indented four spaces because it is nested within two elements: html and head.

1. **Click in the blank line above the closing head tag if necessary, press [Spacebar] four times, then type <title>Lakeland Reeds Bed and Breakfast</title>**
   The title element specifies text that appears in the title bar of the Web browser opening the page. This element is part of the document's head section because the text does not appear in the main browser window.
2. **Click in the blank line between the opening and closing body tags, press [Spacebar] four times, then type <h1>Lakeland Reeds Bed and Breakfast</h1>**
   The h1 element represents the highest-level heading on the page.
3. **Press [Enter], press [Spacebar] four times, then type A country getaway perfect for fishing, boating, biking, or just watching the day go by. **
   The p element marks a paragraph of text.
4. **Press [Enter], then type the following text, pressing [Spacebar] four times at the beginning of each line and pressing [Enter] at the end of each line except the last:**

   ** Philip Blaine, Proprietor**
   45 Marsh Grass Ln.
   Marble, MN 55764
   **(218) 555-5253 **

   Your document should look like the one shown in Figure B-6.
5. **Press [Enter], type , type your first and last name, type a comma, then type HTML5 Unit B **
6. **Save your work**

Adding comments to your HTML code

In addition to marking text that appears on your Web pages, you can create text elements in your Web page code that user agents ignore. These elements, known as **comments**, are not rendered by user agents and are viewable only by people who examine the HTML code of your Web pages. Comments can be especially helpful when you are creating or adding on to a large, complex Web document or Web site, or when other Web developers will be working with your code—now or in the future. Common uses for comments include explaining what a particular section of HTML does or pointing out the beginning and end of parts of a Web page containing numerous HTML elements. The comment tag pair begins with <!-- and ends with -->.

FIGURE B-5: Sketch of the Web page

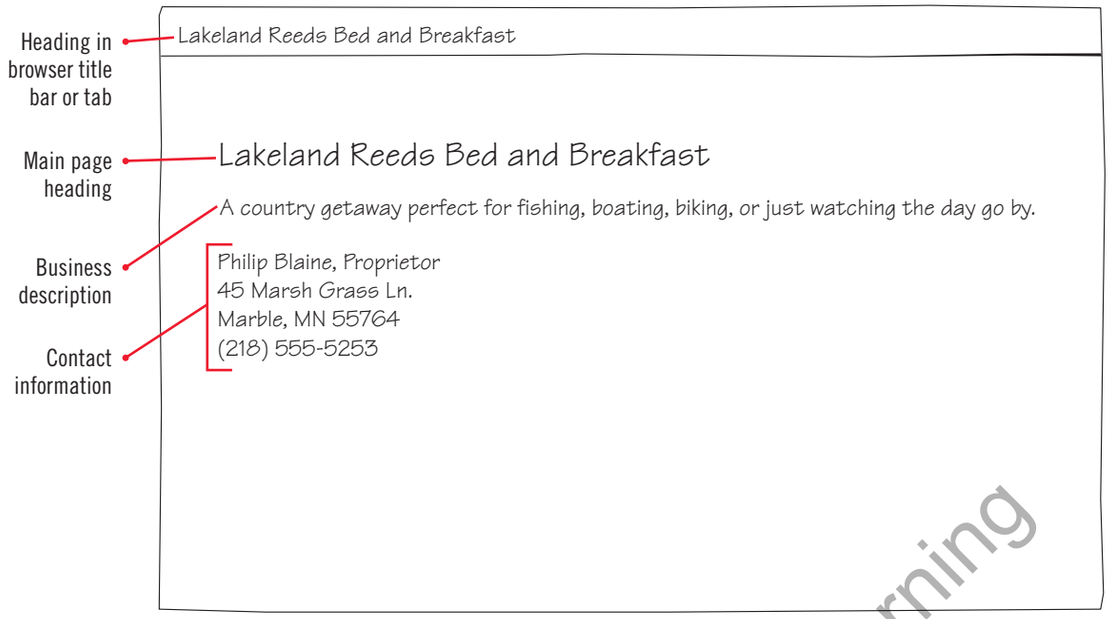
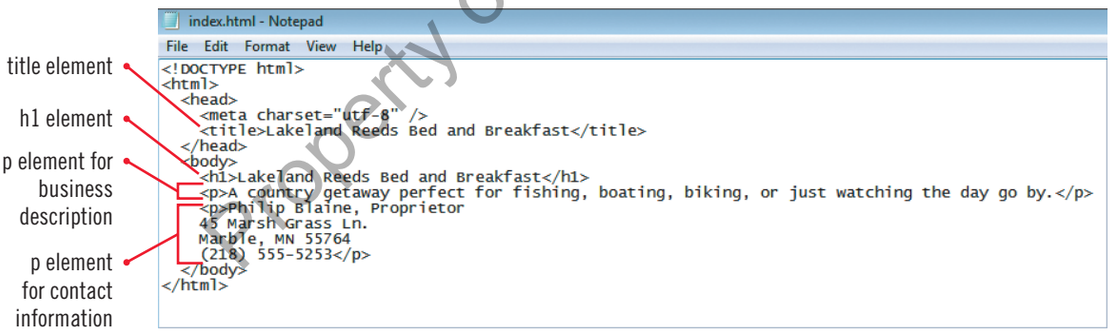


FIGURE B-6: Title, h1, and p elements entered



UNIT
B
HTML5 and CSS3

Previewing Your Web Page

An important step in the process of creating a Web page is to **preview** it, which involves opening it in one or more user agents and examining the result. When a Web page isn't interpreted as expected by a user agent, you can research the problem and make corrections to the code before publishing the page. In addition, because different user agents can interpret the same page with slight differences, it's good practice to test your pages with multiple user agents.  You decide to preview your Lakeland Reeds Web page in a browser.

STEPS

TROUBLE
If your Web page does not match Figure B-6, return to Notepad, compare your code to Figure B-7, edit as necessary, and save the file, then repeat Step 2 to preview your edited Web page.

1. **Using your file manager, navigate to the drive and folder where you store your Data Files, open the Unit B/Unit folder, then double-click index.html**
   The Web page opens in your system's default Web browser, as shown in Figure B-7. The address information runs together on a single line, rather than appearing on multiple lines as in the code.
2. **If multiple browsers are installed on your system, return to your file manager, right-click (Windows) or control-click (Mac) index.html, point to Open with, then click another browser name in the list**
   The Web page opens in a non-primary browser.
3. **If your Web page is open in two browsers, note any differences between them in the way the page is displayed**
   Often differences in the way each browser displays, or **renders**, a Web page are subtle, such as slight variations in the space between lines. Figure B-8 shows a Web page that is displayed differently in multiple browsers.

Why browsers display Web pages differently

The display of Web pages in HTML5 starts with the standards created by the W3C and the WHATWG. The standards list and describe all the available elements, along with parameters for how user agents should handle them. User agents are built around software known as **rendering engines** that translate Web page elements into visual, auditory, or tactile representations based on these standards. Because the standards require some interpretation, no two engines render the same HTML code in exactly the same way. In addition, the creators of rendering engines do not always implement all of the current standards in their software. Because the audience for your Web pages will almost always be using a number of different user agents, it's important to test your code in a variety of popular browsers and on multiple operating systems (such as Windows 7, Windows XP, and Mac OS X).

FIGURE B-7: Previewing the Web page

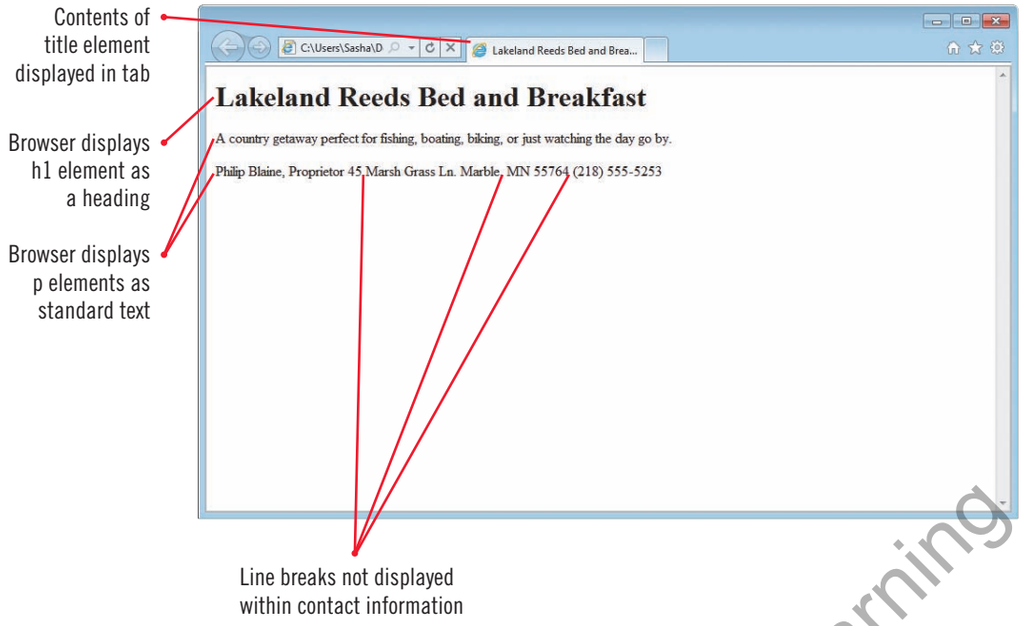
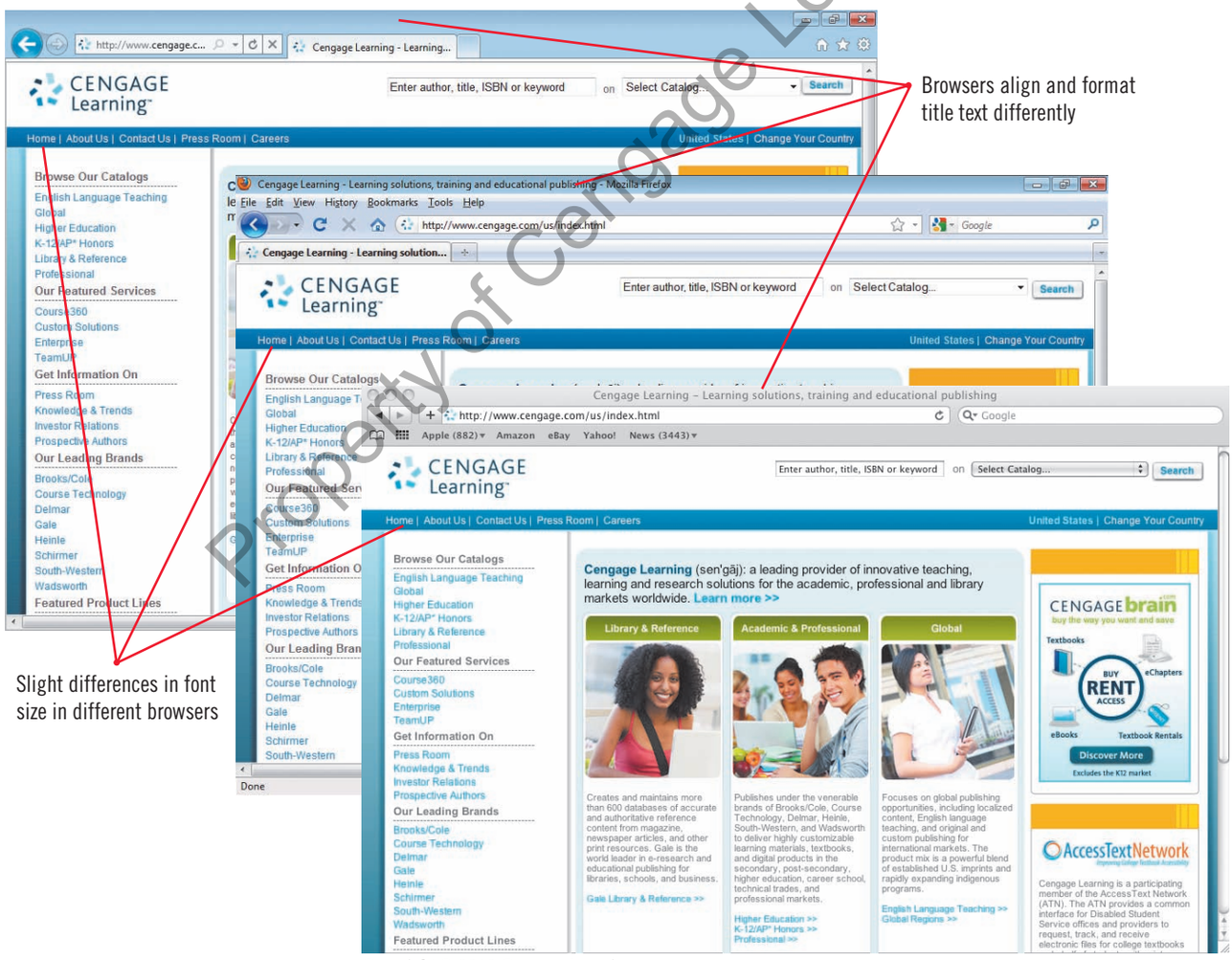


FIGURE B-8: Rendering differences between browsers



UNIT
B
HTML5 and CSS3

Implementing One-Sided Tags

While you add most elements to a Web page using tag pairs, some elements require only a single tag. Instead of affecting text or other elements on the page, these one-sided elements generally represent a specific occurrence of an item or a behavior. You enter a one-sided element by typing a single tag.  Although pressing [Enter] to insert line breaks in your HTML code is useful for making your code easier to understand, user agents ignore line breaks inserted this way when processing HTML code. To make the Lakeland Reeds address and phone number appear on multiple lines, you decide to add HTML line break codes that browsers will incorporate into the page layout.

STEPS

1. **In your text editor, click at the beginning of the line that reads 45 Marsh Grass Ln., then use → and ← if necessary to position the insertion point immediately to the left of the 4**

QUICK TIP
Be sure to type a space between the r and the /.

2. **Press [Backspace] (Windows) or [delete] (Mac) five times to remove the indentation and move the street address to the end of the previous line, then type
**
   The br element inserts a line break. As a result, the text "45 Marsh Grass Ln." will be displayed on a new line when a user opens the document in a browser. Table B-3 summarizes the Web page elements you have implemented in your Web page.
3. **Click at the beginning of the line that reads Marble, MN 55764, then use → and ← if necessary to position the insertion point immediately to the left of the first M**
4. **Press [Backspace] (Windows) or [delete] (Mac) five times, then type
**
5. **Click at the beginning of the line that reads (218) 555-5253</p>, then use → and ← if necessary to position the insertion point immediately to the left of the (**
6. **Press [Backspace] (Windows) or [delete] (Mac) five times, then type
**
   Your document should look like the one shown in Figure B-9.
7. **Press [Ctrl][S] (Windows) or [command][s] (Mac)**
   Your changes are saved.
8. **Return to your primary Web browser, then click your browser's Refresh or Reload button**
   The browser loads the updated version of the Web page. The address and phone number now appear on multiple lines, as shown in Figure B-10.
9. **If you previously opened index.html in a second browser, refresh or reload the page to see the changes in that browser as well**

FIGURE B-9: HTML document containing line breaks

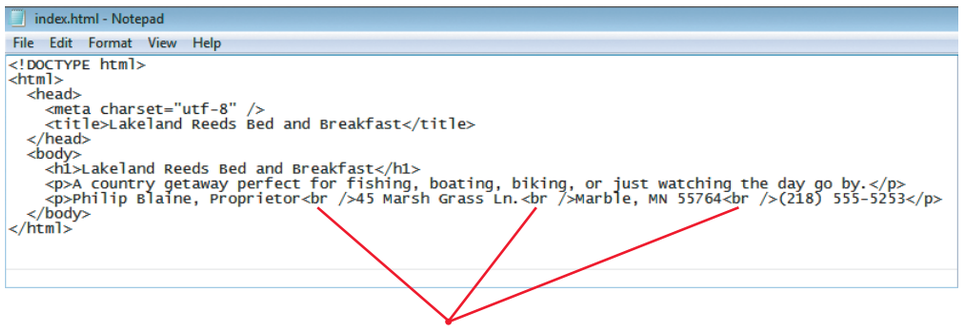


FIGURE B-10: Web page displaying line breaks

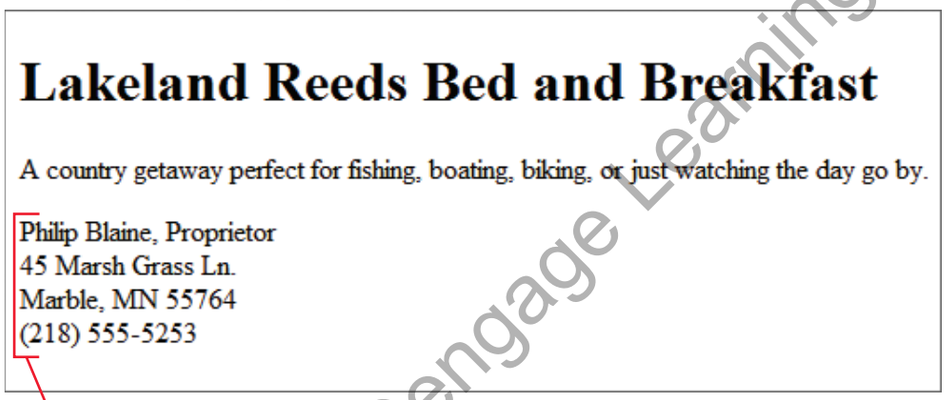


TABLE B-3: Basic Web page elements

element	function	code sample

html	marks the beginning and the end of the Web page	`<html>`
		web page contents
		`</html>`

head	contains elements that are not part of the main Web page	`<head>`
		head contents, such as title and meta elements
		`</head>`

body	includes contents that are visible in the main window of a Web browser	`<body>`
		body contents, such as p and h1 elements
		`</body>`

meta	enables you to pass information about a Web page to user agents that open it	`<meta charset="utf-8" />`
title	specifies text that appears in the title bar of the Web browser opening the page	`<title>Lakeland Reeds</title>`
p	marks a paragraph of text	`<p>Escape to the lake!</p>`
h1	represents the highest-level heading on the page	`<h1>Lakeland Reeds</h1>`
br	inserts a line break	`45 Marsh Grass Ln. Marble, MN 55764`

UNIT
B
HTML5 and CSS3

Validating Your HTML Code

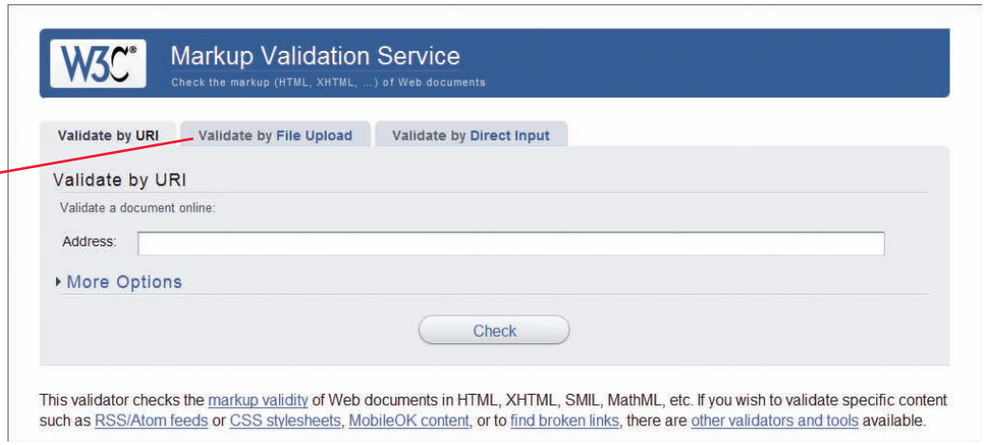
You've seen that previewing your Web pages can be useful for spotting problems with your code and making sure that your Web pages display as expected for your users. Another tool is validation, which is an automated process of comparing code you've written against the HTML5 coding standards. When previewing a page reveals an error in your code that's difficult to track down, validation can sometimes be useful in identifying the specific source of the problem. In addition, sometimes a user agent can interpret a Web page as expected in spite of code that doesn't conform to specifications. In this case, validating your code and correcting errors can help to ensure that your code will continue to work with future versions of both user agents and HTML standards, which may not continue to deal seamlessly with erroneous coding.  Faduma has given you the Web address of an online code validator and asks you to give her a validation report of the page you've created.

STEPS

1. **Open your Web browser, click in the Address Bar, type http://validator.w3.org/, then press [Enter]**
   The Web page opens for the validation service provided by the W3C, as shown in Figure B-11.
2. **Click the Validate by File Upload tab**
   Because your page is not yet published on the Internet, you'll upload your file directly to the w3.org Web site for validation.
3. **Click Browse or Choose File, navigate to the drive and folder where you store your Data Files, open the Unit B/Unit folder, then double-click index.html**
4. **Click Check**
   The browser uploads your document to the w3.org Web site and the result page opens, as shown in Figure B-12.
5. **Scroll down to read the validation results, including any Notes or Warnings**
6. **If you have space on a Web server for publishing your documents, create a local root folder within the Unit folder, move index.html to the local root folder, upload the file to your Web publishing location, then open index.html in your browser from the published location to verify that the upload was successful**
7. **Close any open browsers, then close your text editor.**

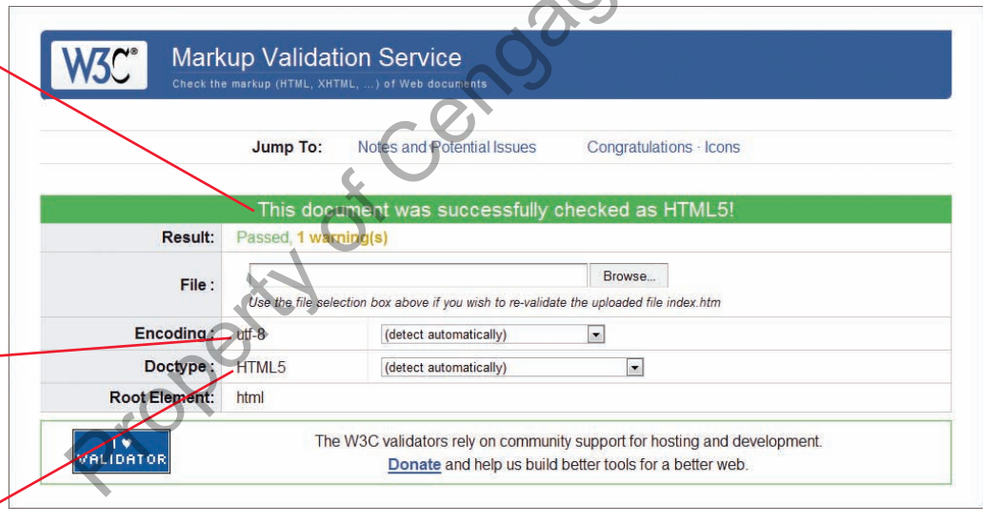
TROUBLE
If your document does not successfully validate as HTML5, return to your text editor, compare your document to Figure B-9 and make any necessary changes, then save your work and revalidate your file.

FIGURE B-11: W3C Validator Web page



Click for options to upload your html file

FIGURE B-12: Validation results Web page



Successful validation means the index.htm code meets the guidelines set out in the HTML5 specification

Character encoding recognized in the meta tag you entered

Doctype detected from your DOCTYPE declaration