Introduction to XHTML

Objectives

- To understand important components of XHTML documents.
- To use XHTML to create World Wide Web pages.
- To add images to Web pages.
- To understand how to create and use hyperlinks to navigate Web pages.
- To mark up lists of information.
- To create forms.

To read between the lines was easier than to follow the text.
Aristophanes

Yea, from the table of my memory
I'll wipe away all trivial fond records.
William Shakespeare

Outline

E.1 Introduction

In this appendix, we introduce *XHTML*[1]—the *Extensible HyperText Markup Language* for creating Web content. Unlike procedural programming languages such as C, Fortran, Cobol and Visual Basic, XHTML is a *markup language* that specifies the format of text that is displayed in a Web browser, such as Microsoft's Internet Explorer or Netscape's Communicator.

One key issue when using XHTML is the separation of the *presentation of a document* (i.e., the document's appearance when rendered by a browser) from the *structure of the document's information*. Throughout this appendix, we will discuss this issue in depth.

In this appendix, we build several complete Web pages featuring text, hyperlinks, images, horizontal rules and line breaks. We also discuss more substantial XHTML features, including presentation of information in *tables* and *incorporating forms* for collecting information from a Web-page visitor. By the end of this appendix, you will be familiar with the most commonly used XHTML features and will be able to create more complex Web documents.

E.2 Editing XHTML

In this appendix, we write XHTML in its *source-code form*. We create *XHTML documents* by typing them in with a text editor (e.g., *Notepad*, *Wordpad*, *vi* or *emacs*) and saving the documents with either an **`.html`** or **`.htm`** file-name extension.

1. XHTML has replaced the HyperText Markup Language (HTML) as the primary means of describing Web content. XHTML provides more robust, richer and more extensible features than HTML. For more on XHTML/HTML, visit **`www.w3.org/markup`**.

Good Programming Practice E.1

Assign documents file names that describe their functionality. This practice can help you identify documents faster. It also helps people who want to link to a page, by giving them an easy-to-remember name. For example, if you are writing an XHTML document that contains product information, you might want to call it ***products.html****.*

Machines running specialized software called a *Web server* store XHTML documents. Clients (e.g., Web browsers) request specific *resources*, such as XHTML documents, from the Web server. For example, typing **www.deitel.com/books/downloads.htm** into a Web browser's address field requests **downloads.htm** from the Web server running at **www.deitel.com**. This document is located in a directory named **books**.

E.3 First XHTML Example

In this appendix, we present XHTML markup and provide screen captures that show how Internet Explorer renders (i.e., displays) the XHTML. Every XHTML document we show has line numbers for the reader's convenience. These line numbers are not part of the XHTML documents.

Our first example (Fig. E.1) is an XHTML document named **main.html** that displays the message **Welcome to XHTML!** in the browser. The key line in the program is line 14, which tells the browser to display **Welcome to XHTML!** Now let us consider each line of the program.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.1: main.html -->
6  <!-- Our first Web page. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Our first Web page</title>
11    </head>
12
13    <body>
14       <p>Welcome to XHTML!</p>
15    </body>
16 </html>
```

Fig. E.1 First XHTML example.

Lines 1–3 are required in XHTML documents to conform with proper XHTML syntax. Lines 5–6 are *XHTML comments*. XHTML document creators insert comments to improve markup readability and to describe the content of a document. Comments also help other people read and understand an XHTML document's markup and content. Comments do not cause the browser to perform any action when the user loads the XHTML document into the Web browser to view the document. XHTML comments always start with **`<!--`** and end with **`-->`**. Each of our XHTML examples includes comments that specify the figure number and file name and provide a brief description of the example's purpose. Subsequent examples include comments in the markup, especially to highlight new features.

Good Programming Practice E.2

Place comments throughout your markup. Comments help other programmers understand the markup, assist in debugging and list useful information that you do not want the browser to render. Comments also help you understand your own markup when you revisit a document for modifications or updates in the future.

XHTML markup contains text that represents the content of a document and *elements* that specify a document's structure. Some important elements of an XHTML document include the **`html`** element, the **`head`** element and the **`body`** element. The **`html`** element encloses the *head section* (represented by the **`head`** *element*) and the *body section* (represented by the **`body`** *element*). The head section contains information about the XHTML document, such as the *title* of the document. The head section also can contain special document-formatting instructions called *style sheets* and client-side programs called *scripts* for creating dynamic Web pages. The body section contains the page's content that the browser displays when the user visits the Web page.

XHTML documents delimit an element with *start* and *end* tags. A start tag consists of the element name in angle brackets (e.g., **`<html>`**). An end tag consists of the element name preceded by a **`/`** in angle brackets (e.g., **`</html>`**). In this example, lines 8 and 16 define the start and end of the **`html`** element. Note that the end tag on line 16 has the same name as the start tag, but is preceded by a **`/`** inside the angle brackets. Many start tags define *attributes* that provide additional information about an element. Browsers can use this additional information to determine how to process the element. Each attribute has a *name* and a *value*, separated by an equal sign (**`=`**). Line 8 specifies a required attribute (**`xmlns`**) and value (**`http://www.w3.org/1999/xhtml`**) for the **`html`** element in an XHTML document.

Common Programming Error E.1

Not enclosing attribute values in either single or double quotes is a syntax error.

Common Programming Error E.2

Using uppercase letters in an XHTML element or attribute name is a syntax error.

An XHTML document divides the **`html`** element into two sections—head and body. Lines 9–11 define the Web page's head section with a **`head`** element. Line 10 specifies a **`title`** element. This is called a *nested element*, because it is enclosed in the **`head`** element's start and end tags. The **`head`** element also is a nested element, because it is enclosed in the **`html`** element's start and end tags. The **`title`** element describes the Web page. Titles usually appear in the *title bar* at the top of the browser window and also as the text

identifying a page when users add the page to their list of **Favorites** or **Bookmarks**, which enable users to return to their favorite sites. Search engines (i.e., sites that allow users to search the Web) also use the `title` for cataloging purposes.

Good Programming Practice E.3

Indenting nested elements emphasizes a document's structure and promotes readability.

Common Programming Error E.3

XHTML does not permit tags to overlap—a nested element's end tag must appear in the document before the enclosing element's end tag. For example, the nested XHTML tags `<head><title>hello</head></title>` *cause a syntax error, because the enclosing* `head` *element's ending* `</head>` *tag appears before the nested* `title` *element's ending* `</title>` *tag.*

Good Programming Practice E.4

Use a consistent `title` *naming convention for all pages on a site. For example, if a site is named "Bailey's Web Site," then the* `title` *of the main page might be "Bailey's Web Site—Links." This practice can help users better understand the Web site's structure.*

Line 13 opens the document's `body` element. The body section of an XHTML document specifies the document's content, which may include text and tags.

Some tags, such as the *paragraph tags* (`<p>` and `</p>`) in line 14, mark up text for display in a browser. All text placed between the `<p>` and `</p>` tags form one paragraph. When the browser renders a paragraph, a blank line usually precedes and follows paragraph text.

This document ends with two closing tags (lines 15–16). These tags close the **body** and `html` elements, respectively. The ending `</html>` tag in an XHTML document informs the browser that the XHTML markup is complete.

To view this example in Internet Explorer, perform the following steps:

1. Copy the Appendix E examples onto your machine (these examples are available on the CD-ROM that accompanies this book).
2. Launch Internet Explorer, and select **Open...** from the **File** Menu. This displays the **Open** dialog.
3. Click the **Open** dialog's **Browse...** button to display the **Microsoft Internet Explorer** file dialog.
4. Navigate to the directory containing the Appendix E examples, and select the file `main.html`; then click **Open**.
5. Click **OK** to have Internet Explorer (or any other browser) render the document. Other examples are opened in a similar manner.

At this point, your browser window should appear similar to the sample screen capture shown in Fig. E.1.

E.4 Headers

Some text in an XHTML document might be more important than other text. For example, the text in this section is considered more important than a footnote. XHTML provides six *headers*, called *header elements*, for specifying the relative importance of information. Figure E.2 demonstrates these elements (**h1** through **h6**).

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.2: header.html -->
6  <!-- XHTML headers.        -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>XHTML headers</title>
11    </head>
12
13    <body>
14
15       <h1>Level 1 Header</h1>
16       <h2>Level 2 header</h2>
17       <h3>Level 3 header</h3>
18       <h4>Level 4 header</h4>
19       <h5>Level 5 header</h5>
20       <h6>Level 6 header</h6>
21
22    </body>
23 </html>
```

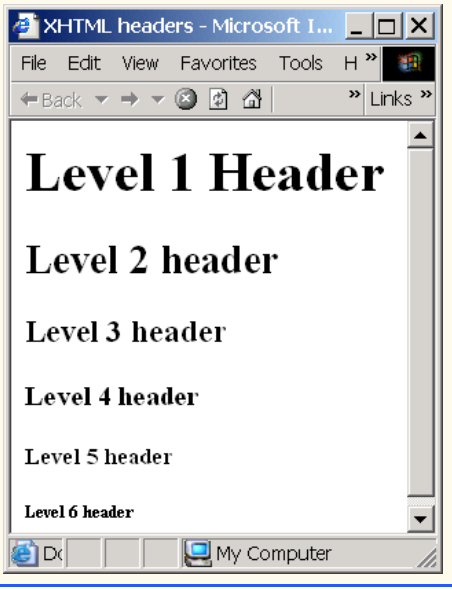


Fig. E.2 Header elements **h1** through **h6**.

Header element **h1** (line 15) is considered the most significant header and is rendered in a larger font than the other five headers (lines 16–20). Each successive header element (i.e., **h2**, **h3**, etc.) is rendered in a smaller font.

Portability Tip E.1

The text size used to display each header element can vary significantly between browsers.

Look-and-Feel Observation E.1

Placing a header at the top of every XHTML page helps viewers understand the purpose of each page.

Look-and-Feel Observation E.2

Use larger headers to emphasize more important sections of a Web page.

E.5 Linking

One of the most important XHTML features is the *hyperlink*, which references (or *links* to) other resources, such as XHTML documents and images. In XHTML, both text and images can act as hyperlinks. Web browsers typically underline text hyperlinks and color their text blue by default, so that users can distinguish hyperlinks from plain text. In Fig. E.3, we create text hyperlinks to four different Web sites. Line 17 introduces the **`<strong>`** tag. Browsers typically display text marked up with **`<strong>`** in a bold font.

Links are created using the **`a`** (*anchor*) *element*. Line 21 defines a hyperlink that links the text **`Deitel`** to the URL assigned to attribute **`href`**, which specifies the location of a linked resource, such as a Web page, a file or an e-mail address. This particular anchor element links to a Web page located at **`http://www.deitel.com`**. When a URL does not indicate a specific document on the Web site, the Web server returns a default Web page. This page often is called **`index.html`**; however, most Web servers can be configured to to use any file as the default Web page for the site. (Open **`http://www.deitel.com`** in one browser window and **`http://www.deitel.com/index.html`** in a second browser window to confirm that they are identical.) If the Web server cannot locate a requested document, the server returns an error indication to the Web browser, and the browser displays an error message to the user.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.3: links.html        -->
6  <!-- Introduction to hyperlinks. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Introduction to hyperlinks</title>
11    </head>
12
13    <body>
14
15       <h1>Here are my favorite sites</h1>
16
17       <p><strong>Click a name to go to that page.</strong></p>
18
```

Fig. E.3 Linking to other Web pages. (Part 1 of 2.)

```
19         <!-- create four text hyperlinks -->
20         <p>
21            <a href = "http://www.deitel.com">Deitel</a>
22         </p>
23
24         <p>
25            <a href = "http://www.prenhall.com">Prentice Hall</a>
26         </p>
27
28         <p>
29            <a href = "http://www.yahoo.com">Yahoo!</a>
30         </p>
31
32         <p>
33            <a href = "http://www.usatoday.com">USA Today</a>
34         </p>
35
36      </body>
37   </html>
```

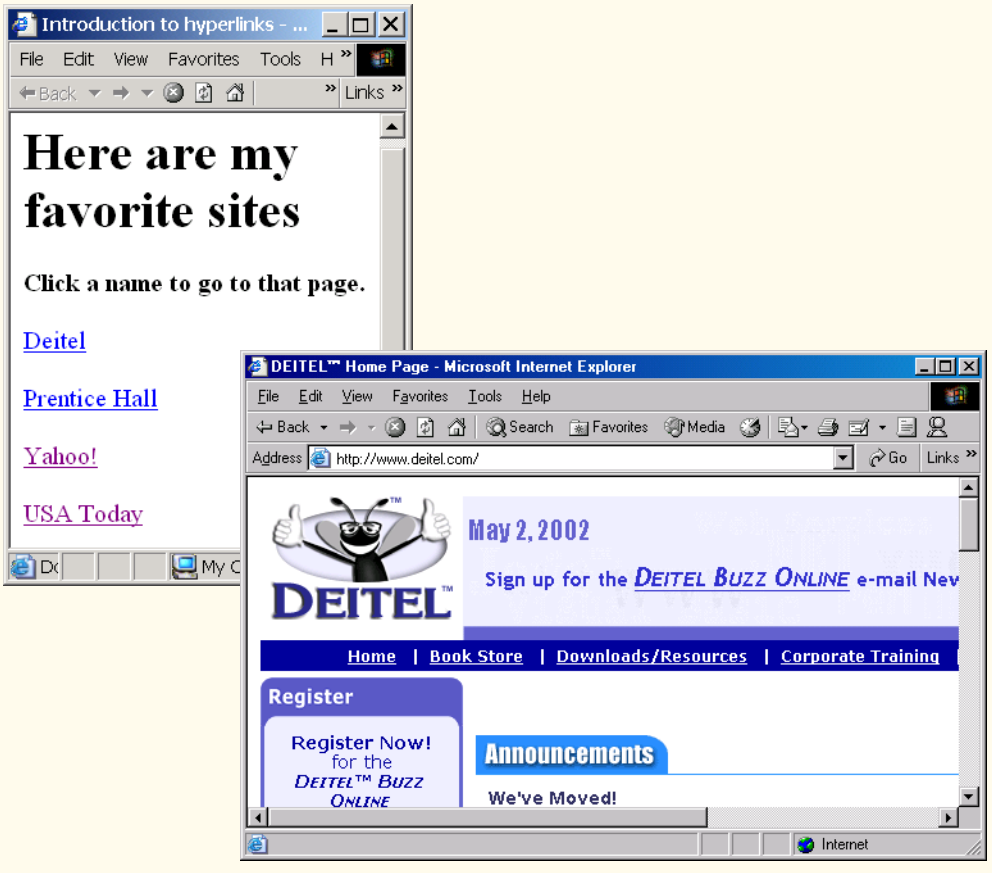


Fig. E.3 Linking to other Web pages. (Part 2 of 2.)

Anchors can link to e-mail addresses through a ***mailto:*** URL. When someone clicks this type of anchored link, most browsers launch the default e-mail program (e.g., Outlook Express) to enable the user to write an e-mail message to the linked address. Figure E.4 demonstrates this type of anchor.

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. E.4: contact.html   -->
<!-- Adding email hyperlinks. -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Adding e-mail hyperlinks</title>
   </head>

   <body>

      <p>My email address is
         <a href = "mailto:deitel@deitel.com">
            deitel@deitel.com
         </a>
         . Click the address and your browser will
         open an e-mail message and address it to me.
      </p>
   </body>
</html>
```

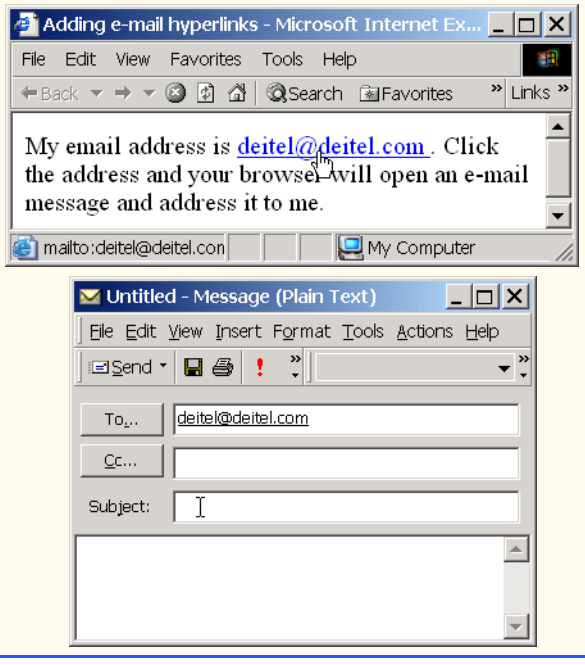


Fig. E.4 Linking to an e-mail address.

Lines 17–19 contain an e-mail link. The form of an e-mail anchor is **...**. In this case, we link to the e-mail address **deitel@deitel.com**.

E.6 Images

The examples discussed so far demonstrated how to mark up documents that contain only text. However, most Web pages contain both text and images. In fact, images are an equal and essential part of Web-page design. The two most popular image formats used by Web developers are Graphics Interchange Format (GIF) and Joint Photographic Experts Group (JPEG) images. Users can create images, using specialized pieces of software, such as Adobe PhotoShop Elements and Jasc Paint Shop Pro (**www.jasc.com**). Images may also be acquired from various Web sites, such as **gallery.yahoo.com**. Figure E.5 demonstrates how to incorporate images into Web pages.

Good Programming Practice E.5

*Always include the **width** and the **height** of an image inside the **<img>** tag. When the browser loads the XHTML file, it will know immediately from these attributes how much screen space to provide for the image and will lay out the page properly, even before it downloads the image.*

Performance Tip E.1

*Including the **width** and **height** attributes in an **<img>** tag will help the browser load and render pages faster.*

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.5: picture.html    -->
6  <!-- Adding images with XHTML. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Adding images in XHTML</title>
11    </head>
12
13    <body>
14
15       <p>
16          <img src = "cool8se.jpg" height = "238" width = "181"
17             alt = "An imaginary landscape." />
18
19          <img src = "fish.jpg" height = "238" width = "181"
20             alt = "A picture of a fish swimming." />
21       </p>
22
23    </body>
24 </html>
```

Fig. E.5 Placing images in XHTML files. (Part 1 of 2.)

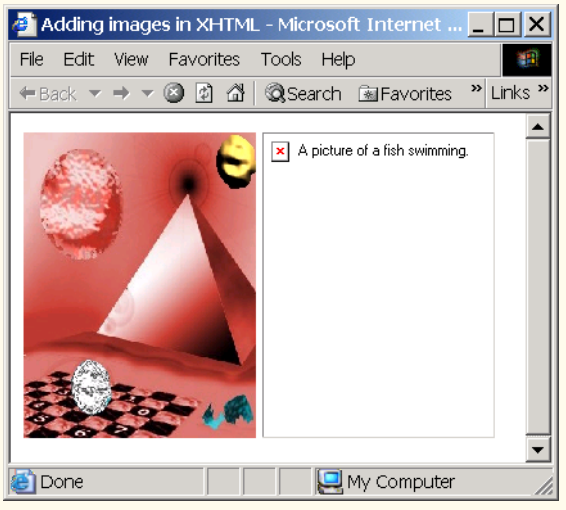


Fig. E.5 Placing images in XHTML files. (Part 2 of 2.)

Common Programming Error E.4

Entering new dimensions for an image that change its inherent width-to-height ratio might distort the appearance of the image. For example, if your image is 200 pixels wide and 100 pixels high, you should ensure that any new dimensions have a 2:1 width-to-height ratio.

Lines 16–17 use an **`img`** element to insert an image in the document. The image file's location is specified with the **`img`** element's **`src`** attribute. In this case, the image is located in the same directory as this XHTML document, so only the image's file name is required. Optional attributes **`width`** and **`height`** specify the image's width and height, respectively. The document author can scale an image by increasing or decreasing the values of the image **`width`** and **`height`** attributes. If these attributes are omitted, the browser uses the image's actual width and height. Images are measured in *pixels* ("picture elements"), which represent dots of color on the screen. The image in Fig. E.5 is **`181`** pixels wide and **`238`** pixels high.

Every **`img`** element in an XHTML document has an **`alt`** attribute. If a browser cannot render an image, the browser displays the **`alt`** attribute's value. A browser might not be able to render an image for several reasons. It might not support images—as is the case with a *text-based browser* (i.e., a browser that can display only text)—or the client may have disabled image viewing to reduce download time. Figure E.5 shows Internet Explorer rendering the **`alt`** attribute's value when a document references a nonexistent image file (**`fish.jpg`**).

The **`alt`** attribute is important for creating *accessible* Web pages for users with disabilities, especially those with vision impairments and text-based browsers. Specialized software called a *speech synthesizer* often is used by people with disabilities. Such software applications "speak" the **`alt`** attribute's value so that the user knows what the browser is displaying.

Some XHTML elements (called *empty elements*) contain only attributes and do not mark up text (i.e., text is not placed between the start and end tags). Empty elements (e.g., **img**) must be terminated, either by using the *forward slash character* (/) inside the closing right angle bracket (**>**) of the start tag or by explicitly including the end tag. When using the forward slash character, we add a space before the forward slash to improve readability (as shown at the ends of lines 17 and 20). Rather than using the forward slash character, lines 19–20 could be written with a closing **</img>** tag as follows:

```
<img src = "cool8se.jpg" height = "238" width = "181"
   alt = "An imaginary landscape."></img>
```

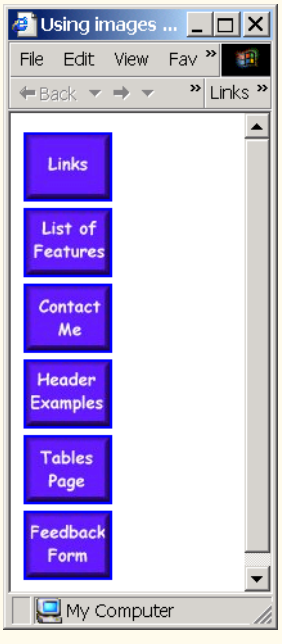
By using images as hyperlinks, Web developers can create graphical Web pages that link to other resources. In Fig. E.6, we create six different image hyperlinks.

Lines 16–19 create an *image hyperlink* by nesting an **img** element within an anchor (**a**) element. The value of the **img** element's **src** attribute value specifies that this image (**links.jpg**) resides in a directory named **buttons**. The **buttons** directory and the XHTML document are in the same directory. Images from other Web documents also can be referenced (after obtaining permission from the document's owner) by setting the **src** attribute to the name and location of the image.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.6: nav.html             -->
6  <!-- Using images as link anchors. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Using images as link anchors</title>
11    </head>
12
13    <body>
14
15       <p>
16          <a href = "links.html">
17             <img src = "buttons/links.jpg" width = "65"
18                height = "50" alt = "Links Page" />
19          </a><br />
20
21          <a href = "list.html">
22             <img src = "buttons/list.jpg" width = "65"
23                height = "50" alt = "List Example Page" />
24          </a><br />
25
26          <a href = "contact.html">
27             <img src = "buttons/contact.jpg" width = "65"
28                height = "50" alt = "Contact Page" />
29          </a><br />
30
```

Fig. E.6 Using images as link anchors. (Part 1 of 2.)

```
31            <a href = "header.html">
32               <img src = "buttons/header.jpg" width = "65"
33                  height = "50" alt = "Header Page" />
34            </a><br />
35
36            <a href = "table.html">
37               <img src = "buttons/table.jpg" width = "65"
38                  height = "50" alt = "Table Page" />
39            </a><br />
40
41            <a href = "form.html">
42               <img src = "buttons/form.jpg" width = "65"
43                  height = "50" alt = "Feedback Form" />
44            </a><br />
45         </p>
46
47      </body>
48   </html>
```



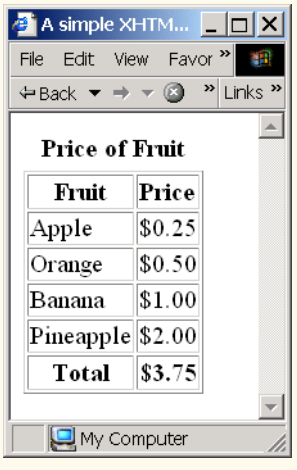


Fig. E.6 Using images as link anchors. (Part 2 of 2.)

On line 19, we introduce the ***br*** *element*, which most browsers render as a *line break*. Any markup or text following a **br** element is rendered on the next line. Like the **img** element, **br** is an example of an empty element terminated with a forward slash. We add a space before the forward slash to enhance readability.

E.7 Special Characters and More Line Breaks

When marking up text, certain characters or symbols (e.g., **<**) might be difficult to embed directly into an XHTML document. Some keyboards do not provide these symbols, or the presence of these symbols could cause syntax errors. For example, the markup

```
<p>if x < 10 then increment x by 1</p>
```

results in a syntax error, because it uses the less-than character (**<**), which is reserved for start tags and end tags such as ** ** and ** **. XHTML provides *special characters* or *entity references* (in the form **&***code***;**) for representing these characters. We could correct the previous line by writing

```
<p>if x &lt; 10 then increment x by 1</p>
```

which uses the special character ***<*** for the less-than symbol.

Figure E.7 demonstrates how to use special characters in an XHTML document. Lines 26–27 contain other special characters, which are expressed either as word abbreviations (e.g., **&** for ampersand and **©** for copyright) or as *hexadecimal (hex)* values (e.g., **&** is the hexadecimal representation of **&**). Hexadecimal numbers are base-16 numbers—digits in a hexadecimal number have values from 0 to 15 (a total of 16 different values). The letters A–F represent the hexadecimal digits corresponding to decimal values 10–15. Thus, in hexadecimal notation, we can have numbers like 876 consisting solely of decimal-like digits, numbers like DA19F consisting of digits and letters, and numbers like DCB consisting solely of letters.

In lines 33–35, we introduce three new elements. Most browsers render the ***del*** element as strike-through text. With this format, users can easily indicate document revisions. To *superscript* text (i.e., raise text on a line with a decreased font size) or *subscript* text (i.e., lower text on a line with a decreased font size), use the ***sup*** and ***sub*** elements, respectively. We also use special characters **<** for a less-than sign and ***¼*** for the fraction 1/4 (line 37).

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.7: contact2.html       -->
6  <!-- Inserting special characters. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Inserting special characters</title>
11    </head>
12
13    <body>
14
```

Fig. E.7 Inserting special characters into XHTML. (Part 1 of 2.)

```
15        <!-- special characters are    -->
16        <!-- entered using form &code; -->
17        <p>
18           Click
19           <a href = "mailto:deitel@deitel.com">here
20           </a> to open an e-mail message addressed to
21           deitel@deitel.com.
22        </p>
23
24        <hr /> <!-- inserts a horizontal rule -->
25
26        <p>All information on this site is <strong>&copy;</strong>
27           Deitel <strong>&amp;</strong> Associates, Inc. 2003.</p>
28
29        <!-- to strike through text use <del> tags    -->
30        <!-- to subscript text use <sub> tags         -->
31        <!-- to superscript text use <sup> tags       -->
32        <!-- these tags are nested inside other tags -->
33        <p><del>You may download 3.14 x 10<sup>2</sup>
34           characters worth of information from this site.</del>
35           Only <sub>one</sub> download per hour is permitted.</p>
36
37        <p>Note: <strong>&lt; &frac14;</strong> of the information
38           presented here is updated daily.</p>
39
40     </body>
41  </html>
```

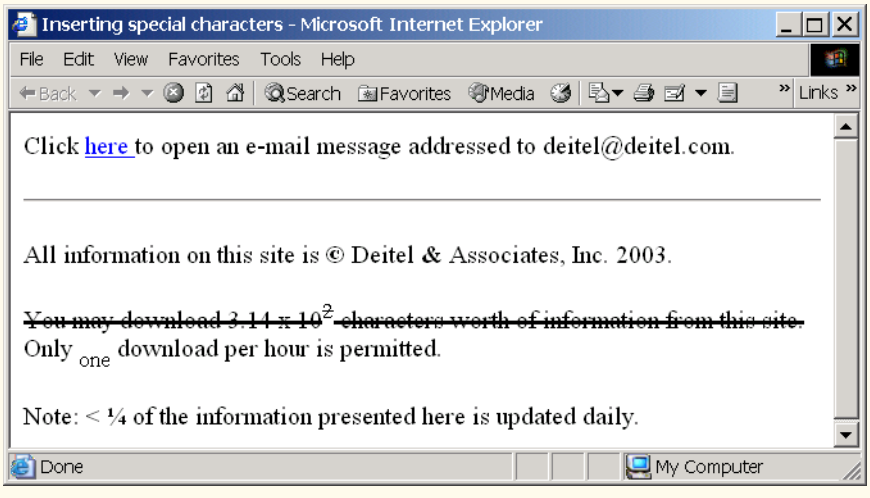


Fig. E.7 Inserting special characters into XHTML. (Part 2 of 2.)

In addition to special characters, this document introduces a *horizontal rule*, indicated by the **<hr />** tag in line 24. Most browsers render a horizontal rule as a horizontal line. The **<hr />** tag also inserts a line break above and below the horizontal line.

E.8 Unordered Lists

Up to this point, we have presented basic XHTML elements and attributes for linking to resources, creating headers, using special characters and incorporating images. In this section, we discuss how to organize information on a Web page using lists. Later in the appendix, we introduce another feature for organizing information, called a table. Figure E.8 displays text in an *unordered list* (i.e., a list that does not order its items by letter or number). The *unordered list element* **`ul`** creates a list in which each item begins with a bullet (called a *disc*).

Each entry in an unordered list (element **`ul`** in line 20) is an **`li`** (*list item*) element (lines 23, 25, 27 and 29). Most Web browsers render these elements with a line break and a bullet symbol indented from the beginning of the new line.

E.9 Nested and Ordered Lists

Lists may be nested to represent hierarchical relationships, as in an outline format. Figure E.9 demonstrates nested lists and *ordered lists* (i.e., list that order their items by letter or number).

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.8: links2.html                  -->
6  <!-- Unordered list containing hyperlinks. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Unordered list containing hyperlinks</title>
11    </head>
12
13    <body>
14
15       <h1>Here are my favorite sites</h1>
16
17       <p><strong>Click on a name to go to that page.</strong></p>
18
19       <!-- create an unordered list -->
20       <ul>
21
22          <!-- add four list items -->
23          <li><a href = "http://www.deitel.com">Deitel</a></li>
24
25          <li><a href = "http://www.w3.org">W3C</a></li>
26
27          <li><a href = "http://www.yahoo.com">Yahoo!</a></li>
28
29          <li><a href = "http://www.cnn.com">CNN</a></li>
30
31        </ul>
32
33    </body>
34 </html>
```

Fig. E.8 Unordered lists in XHTML. (Part 1 of 2.)

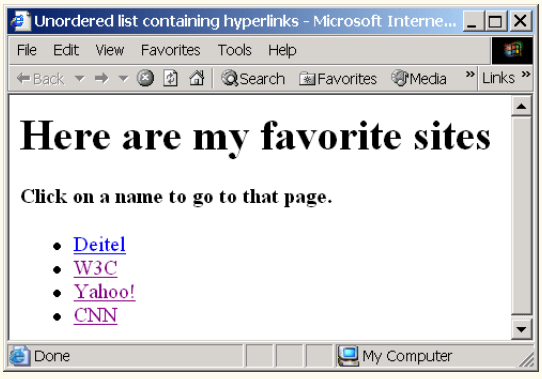


Fig. E.8 Unordered lists in XHTML. (Part 2 of 2.)

The first ordered list begins in line 33. Attribute **`type`** specifies the *sequence type* (i.e., the set of numbers or letters used in the ordered list). In this case, setting **`type`** to **`"I"`** specifies upper-case roman numerals. Line 47 begins the second ordered list and sets attribute **`type`** to **`"a"`**, specifying lowercase letters for the list items. The last ordered list (lines 71–75) does not use attribute **`type`**. By default, the list's items are enumerated from one to three.

A Web browser indents each nested list to indicate a hierarchal relationship. By default, the items in the outermost unordered list (line 18) are preceded by *discs*. List items nested inside the unordered list of line 18 are preceded by *circles*. Although not demonstrated in this example, subsequent nested list items are preceded by *squares*. Unordered list items can be explicitly set to discs, circles or squares by setting the **`ul`** element's **`type`** attribute to **`"disc"`**, **`"circle"`** or **`"square"`**, respectively.

E.10 Basic XHTML Tables

This section presents the XHTML *table*—a frequently used feature that organizes data into rows and columns. Our first example (Fig. E.10) uses a table with six rows and two columns to display price information for fruit.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
4
5  <!-- Fig. E.9: list.html                  -->
6  <!-- Advanced Lists: nested and ordered. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Advanced lists</title>
11    </head>
12
13    <body>
14
```

Fig. E.9 Nested and ordered lists in XHTML. (Part 1 of 3.)

```
15         <h1>The Best Features of the Internet</h1>
16
17         <!-- create an unordered list -->
18         <ul>
19            <li>You can meet new people from countries around
20               the world.</li>
21
22            <li>
23               You have access to new media as it becomes public:
24
25               <!-- start nested list, use modified bullets -->
26               <!-- list ends with closing </ul> tag        -->
27               <ul>
28                  <li>New games</li>
29                  <li>
30                     New applications
31
32                     <!-- ordered nested list -->
33                     <ol type = "I">
34                        <li>For business</li>
35                        <li>For pleasure</li>
36                     </ol>
37
38                  </li>
39
40                  <li>Around the clock news</li>
41                  <li>Search engines</li>
42                  <li>Shopping</li>
43                  <li>
44                     Programming
45
46                     <!-- another nested ordered list -->
47                     <ol type = "a">
48                        <li>XML</li>
49                        <li>Java</li>
50                        <li>XHTML</li>
51                        <li>Scripts</li>
52                        <li>New languages</li>
53                     </ol>
54
55                  </li>
56
57               </ul> <!-- ends nested list started in line 27 -->
58
59            </li>
60
61            <li>Links</li>
62            <li>Keeping in touch with old friends</li>
63            <li>It is the technology of the future!</li>
64
65         </ul>   <!-- ends unordered list started in line 18 -->
66
67         <h1>My 3 Favorite <em>CEOs</em></h1>
```

Fig. E.9 Nested and ordered lists in XHTML. (Part 2 of 3.)

```
68
69          <!-- ol elements without type attribute have -->
70          <!-- numeric sequence type (i.e., 1, 2, ...) -->
71          <ol>
72             <li>Lawrence J. Ellison</li>
73             <li>Steve Jobs</li>
74             <li>Michael Dell</li>
75          </ol>
76
77       </body>
78    </html>
```

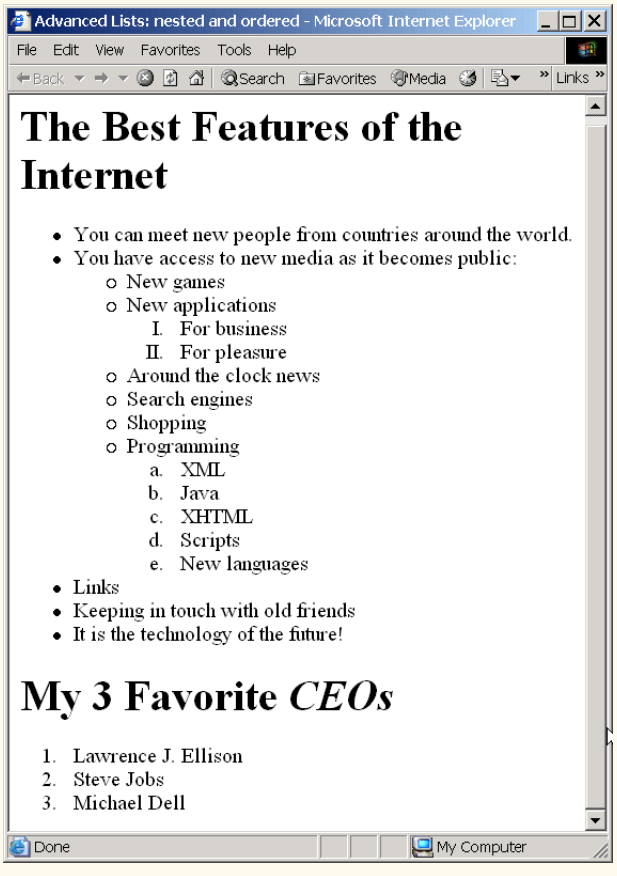


Fig. E.9 Nested and ordered lists in XHTML. (Part 3 of 3.)

Tables are defined with the **`table`** element. Lines 16–18 specify the start tag for a table element that has several attributes. The ***`border`*** attribute specifies the table's border width in pixels. To create a table without a border, set **`border`** to **`"0"`**. This example assigns attribute **`width "40%"`**, to set the table's width to 40 percent of the browser's width. A developer can also set attribute **`width`** to a specified number of pixels.

As its name implies, attribute ***summary*** (line 17) describes the table's contents. Speech devices use this attribute to make the table more accessible to users with visual impairments. The ***caption*** element (line 22) describes the table's content and helps text-based browsers interpret the table data. Text inside the **<caption>** tag is rendered above the table by most browsers. Attribute **summary** and element **caption** are two of many XHTML features that make Web pages more accessible to users with disabilities.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.10: table1.html  -->
6  <!-- Creating a basic table. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Creating a basic table</title>
11    </head>
12
13    <body>
14
15       <!-- the <table> tag begins table -->
16       <table border = "1" width = "40%"
17          summary = "This table provides information about
18             the price of fruit">
19
20          <!-- <caption> tag summarizes table's   -->
21          <!-- contents to help visually impaired -->
22          <caption><strong>Price of Fruit</strong></caption>
23
24          <!-- <thead> is first section of table -->
25          <!-- it formats table header area      -->
26          <thead>
27             <tr> <!-- <tr> inserts one table row -->
28                <th>Fruit</th> <!-- insert heading cell -->
29                <th>Price</th>
30             </tr>
31          </thead>
32
33          <!-- all table content is enclosed within <tbody> -->
34          <tbody>
35             <tr>
36                <td>Apple</td> <!-- insert data cell -->
37                <td>$0.25</td>
38             </tr>
39
40             <tr>
41                <td>Orange</td>
42                <td>$0.50</td>
43             </tr>
44
```

Fig. E.10 XHTML table. (Part 1 of 2.)

```
45             <tr>
46                <td>Banana</td>
47                <td>$1.00</td>
48             </tr>
49
50             <tr>
51                <td>Pineapple</td>
52                <td>$2.00</td>
53             </tr>
54          </tbody>
55
56          <!-- <tfoot> is last section of table -->
57          <!-- it formats table footer          -->
58          <tfoot>
59             <tr>
60                <th>Total</th>
61                <th>$3.75</th>
62             </tr>
63          </tfoot>
64
65       </table>
66
67    </body>
68 </html>
```

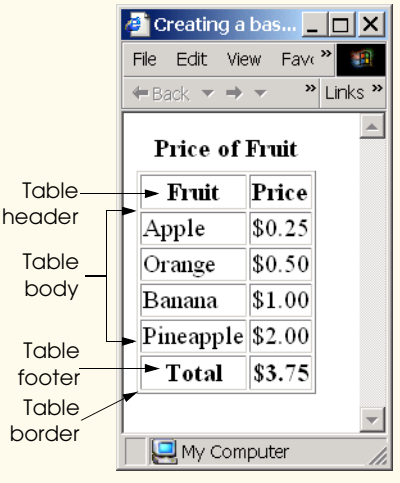


Fig. E.10 XHTML table. (Part 2 of 2.)

Testing and Debugging Tip E.1

Try resizing the browser window to see how the width of the window affects the width of the table.

A table has three distinct sections—*head*, *body* and *foot*. The head section (or *header cell*) is defined with a **`thead`** element (lines 26–31), which contains header information, such as column names. Each **`tr`** element (lines 27–30) defines an individual *table row*. The columns in the head section are defined with **`th`** elements. Most browsers center text for-

matted by **th** (table header column) elements and display it in bold. Table header elements are nested inside table row elements.

The body section, or *table body*, contains the table's primary data. The table body (lines 34–54) is defined in a **tbody** element. *Data cells* contain individual pieces of data and are defined with **td** (*table data*) elements.

The foot section (lines 58–63) is defined with a ***tfoot*** (table foot) element and represents a footer. Text commonly placed in the footer includes calculation results and footnotes. Like other sections, the foot section can contain table rows and each row can contain columns.

E.11 Intermediate XHTML Tables and Formatting

In the previous section, we explored the structure of a basic table. In Fig. E.11, we enhance our discussion of tables by introducing elements and attributes that allow the document author to build more complex tables.

Common Programming Error E.5

*When using **colspan** and **rowspan** to adjust the size of table data cells, keep in mind that the modified cells will occupy more than one column or row; other rows or columns of the table must compensate for the extra rows or columns spanned by individual cells. If you do not, the formatting of your table will be distorted, and you could inadvertently create more columns and rows than you originally intended.*

The table begins on line 17. Element ***colgroup*** (lines 22–27) groups and formats columns. The ***col*** element (line 26) specifies two attributes in this example. The ***align*** attribute determines the alignment of text in the column. The ***span*** attribute determines how many columns the ***col*** element formats. In this case, we set **align**'s value to **"right"** and **span**'s value to **"1"** to right-align text in the first column (the column containing the picture of the camel in the sample screen capture).

Table cells are sized to fit the data they contain. Document authors can create larger data cells by using attributes ***rowspan*** and ***colspan***. The values assigned to these attributes specify the number of rows or columns occupied by a cell. The **th** element at lines 36–39 uses the attribute **rowspan = "2"** to allow the cell containing the picture of the camel to use two vertically adjacent cells (thus the cell *spans* two rows). The **th** element at lines 42–45 uses the attribute **colspan = "4"** to widen the header cell (containing **Camelid comparison** and **Approximate as of 9/2002**) to span four cells.

Line 42 introduces attribute ***valign***, which aligns data vertically and may be assigned one of four values—**"top"** aligns data with the top of the cell, **"middle"** vertically centers data (the default for all data and header cells), **"bottom"** aligns data with the bottom of the cell and **"baseline"** ignores the fonts used for the row data and sets the bottom of all text in the row on a common *baseline* (i.e., the horizontal line to which each character in a word is aligned).

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
```

Fig. E.11 Complex XHTML table. (Part 1 of 3.)

```
<!-- Fig. E.11: table2.html      -->
<!-- Intermediate table design. -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Intermediate table design</title>
   </head>

   <body>

      <h1>Table Example Page</h1>

      <table border = "1">
         <caption>Here is a more complex sample table.</caption>

         <!-- <colgroup> and <col> tags are -->
         <!-- used to format entire columns -->
         <colgroup>

            <!-- span attribute determines how  -->
            <!-- many columns <col> tag affects -->
            <col align = "right" span = "1" />
         </colgroup>

         <thead>

            <!-- rowspans and colspans merge specified    -->
            <!-- number of cells vertically or horizontally   -->
            <tr>

               <!-- merge two rows -->
               <th rowspan = "2">
                  <img src = "camel.gif" width = "205"
                     height = "167" alt = "Picture of a camel" />
               </th>

               <!-- merge four columns -->
               <th colspan = "4" valign = "top">
                  <h1>Camelid comparison</h1><br />
                  <p>Approximate as of 9/2002</p>
               </th>
            </tr>

            <tr valign = "bottom">
               <th># of Humps</th>
               <th>Indigenous region</th>
               <th>Spits?</th>
               <th>Produces Wool?</th>
            </tr>

         </thead>

```

Fig. E.11 Complex XHTML table. (Part 2 of 3.)

```
57          <tbody>
58
59             <tr>
60                <th>Camels (bactrian)</th>
61                <td>2</td>
62                <td>Africa/Asia</td>
63                <td rowspan = "2">Llama</td>
64                <td rowspan = "2">Llama</td>
65             </tr>
66
67             <tr>
68                <th>Llamas</th>
69                <td>1</td>
70                <td>Andes Mountains</td>
71             </tr>
72
73          </tbody>
74
75       </table>
76
77    </body>
78 </html>
```

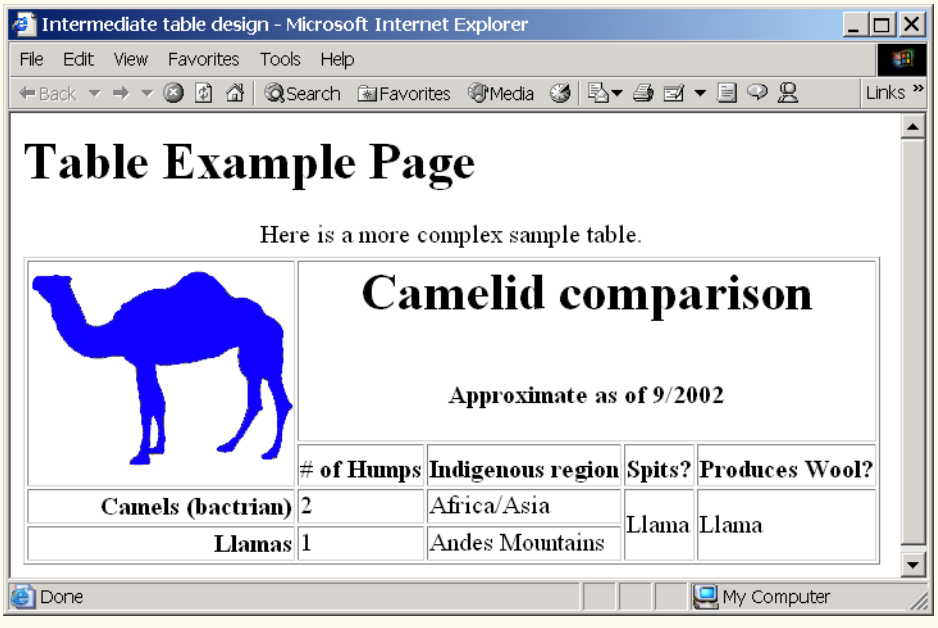


Fig. E.11 Complex XHTML table. (Part 3 of 3.)

E.12 Basic XHTML Forms

When browsing Web sites, users often need to provide information such as e-mail addresses, search keywords and zip codes. XHTML provides a mechanism, called a *form*, for collecting such user information.

Data that users enter on a Web page normally is sent to a Web server that provides access to a site's resources (e.g., XHTML documents or images). These resources are located either on the same machine as the Web server or on a machine that the Web server can access through the network. When a browser requests a Web page or file that is located on a server, the server processes the request and returns the requested resource. A request contains the name and path of the desired resource and the method of communication (called a *protocol*). XHTML documents use the Hypertext Transfer Protocol (HTTP).

Figure E.12 sends the form data to the Web server, which passes the form data to a *form handler*. The form handler processes the data received from the Web server and typically returns information to the Web server. The Web server then sends the information in the form of an XHTML document to the Web browser. [*Note*: This example demonstrates client-side functionality. If the form is submitted (by clicking **Submit Your Entries**), an error occurs.]

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.12: form.html   -->
6  <!-- Form design example 1. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Form design example 1</title>
11    </head>
12
13    <body>
14
15       <h1>Feedback Form</h1>
16
17       <p>Please fill out this form to help
18          us improve our site.</p>
19
20       <!-- <form> tag begins form, gives -->
21       <!-- method of sending information -->
22       <!-- and location of form scripts  -->
23       <form method = "post" action = "/cgi-bin/formmail">
24
25          <p>
26
27             <!-- hidden inputs contain non-visual -->
28             <!-- information                      -->
29             <input type = "hidden" name = "recipient"
30                value = "deitel@deitel.com" />
31
32             <input type = "hidden" name = "subject"
33                value = "Feedback Form" />
```

Fig. E.12 Simple form with hidden fields and a text box. (Part 1 of 2.)

```
34
35             <input type = "hidden" name = "redirect"
36                value = "main.html" />
37          </p>
38
39          <!-- <input type = "text"> inserts text box -->
40          <p>
41             <label>Name:
42                <input name = "name" type = "text" size = "25"
43                   maxlength = "30" />
44             </label>
45          </p>
46
47          <p>
48
49             <!-- input types "submit" and "reset" -->
50             <!-- insert buttons for submitting    -->
51             <!-- and clearing form's contents     -->
52             <input type = "submit" value =
53                "Submit Your Entries" />
54
55             <input type = "reset" value =
56                "Clear Your Entries" />
57          </p>
58
59       </form>
60
61    </body>
62 </html>
```

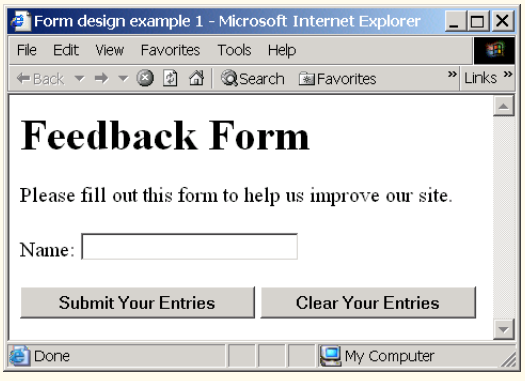


Fig. E.12 Simple form with hidden fields and a text box. (Part 2 of 2.)

Forms can contain visual and non-visual components. Visual components include clickable buttons and other graphical user interface components with which users interact. Non-visual components, called *hidden inputs*, store any data that the document author specifies, such as e-mail addresses and XHTML document file names that act as links. The form begins on line 23 with the **`form`** element. Attribute **`method`** specifies how the form's data is sent to the Web server.

Using **method = "post"** appends form data to the browser request, which contains the protocol (i.e., HTTP) and the requested resource's URL. Scripts located on the Web server's computer (or on a computer accessible through the network) can access the form data sent as part of the request. For example, a script may take the form information and update an electronic mailing list. The other possible value, **method = "get"**, appends the form data directly to the end of the URL. For example, the URL **/cgi-bin/formmail** might have the form information **name = bob** appended to it.

The **action** attribute in the **<form>** tag specifies the URL of a script on the Web server; in this case, it specifies a script that e-mails form data to an address. Most Internet Service Providers (ISPs) have a script like this on their site; ask the Web-site system administrator how to set up an XHTML document to use the script correctly.

Lines 29–36 define three **input** elements that specify data to provide to the script that processes the form (also called the *form handler*). These three **input** elements have **type** attribute **"hidden"**, which allows the document author to send form data that is not entered by a user to a script.

The three hidden inputs are an e-mail address to which the data will be sent, the e-mail's subject line and a URL where the browser will be redirected after submitting the form. Two other **input** attributes are **name**, which identifies the **input** element, and **value**, which provides the value that will be sent (or posted) to the Web server.

Good Programming Practice E.6

Place hidden **input** *elements at the beginning of a form, immediately after the opening* **<form>** *tag. This placement allows document authors to locate hidden* **input** *elements quickly.*

We introduce another **type** of **input** in lines 38–39. The **"text" input** inserts a *text box* into the form. Users can type data in text boxes. The **label** element (lines 37–40) provides users with information about the **input** element's purpose.

Common Programming Error E.6

Forgetting to include a **label** *element for each form element is a design error. Without these labels, users cannot determine the purpose of individual form elements.*

The **input** element's **size** attribute specifies the number of characters visible in the text box. Optional attribute **maxlength** limits the number of characters input into the text box. In this case, the user is not permitted to type more than **30** characters into the text box.

There are two types of **input** elements in lines 52–56. The **"submit" input** element is a button. When the user presses a **"submit"** button, the browser sends the data in the form to the Web server for processing. The **value** *attribute* sets the text displayed on the button (the default value is **Submit**). The **"reset" input** element allows a user to reset all **form** elements to their default values. The **value** attribute of the **"reset" input** element sets the text displayed on the button (the default value is **Reset**).

E.13 More Complex XHTML Forms

In the previous section, we introduced basic forms. In this section, we introduce elements and attributes for creating more complex forms. Figure E.13 contains a form that solicits user feedback about a Web site.

The **`textarea`** element (lines 42–44) inserts a multiline text box, called a *textarea*, into the form. The number of rows is specified with the **`rows`** *attribute* and the number of columns (i.e., characters) is specified with the **`cols`** *attribute*. In this example, the **`textarea`** is four rows high and 36 characters wide. To display default text in the text area, place the text between the **`<textarea>`** and **`</textarea>`** tags. Default text can be specified in other **`input`** types, such as text boxes, by using the **`value`** attribute.

The **`"password"`** input in lines 52–53 inserts a password box with the specified **`size`**. A password box allows users to enter sensitive information, such as credit card numbers and passwords, by "masking" the information input with asterisks. The actual value input is sent to the Web server, not the asterisks that mask the input.

Lines 60–78 introduce the *checkbox* **`form`** element. Checkboxes enable users to select from a set of options. When a user selects a checkbox, a check mark appears in the check box. Otherwise, the checkbox remains empty. Each **`"checkbox"`** **`input`** creates a new checkbox. Checkboxes can be used individually or in groups. Checkboxes that belong to a group are assigned the same **`name`** (in this case, **`"thingsliked"`**).

We continue our discussion of forms by presenting a third example that introduces several more form elements from which users can make selections (Fig. E.14). In this example, we introduce two new **`input`** types. The first type is the *radio button* (lines 90–113), specified with type **`"radio"`**. Radio buttons are similar to checkboxes, except that only one radio button in a group of radio buttons may be selected at any time. All radio buttons in a group have the same **`name`** attribute; they are distinguished by their different **`value`** attributes. The attribute–value pair **`checked = "checked"`** (line 92) indicates which radio button, if any, is selected initially. The **`checked`** attribute also applies to checkboxes.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.13: form2.html  -->
6  <!-- Form design example 2. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Form design example 2</title>
11    </head>
12
13    <body>
14
15       <h1>Feedback Form</h1>
16
17       <p>Please fill out this form to help
18          us improve our site.</p>
19
20       <form method = "post" action = "/cgi-bin/formmail">
21
22          <p>
23             <input type = "hidden" name = "recipient"
24                value = "deitel@deitel.com" />
```

Fig. E.13 Form with textareas, password boxes and checkboxes. (Part 1 of 3.)

```
25
26             <input type = "hidden" name = "subject"
27                value = "Feedback Form" />
28
29             <input type = "hidden" name = "redirect"
30                value = "main.html" />
31          </p>
32
33          <p>
34             <label>Name:
35                <input name = "name" type = "text" size = "25" />
36             </label>
37          </p>
38
39          <!-- <textarea> creates multiline textbox -->
40          <p>
41             <label>Comments:<br />
42                <textarea name = "comments" rows = "4"
43                   cols = "36">Enter your comments here.
44                </textarea>
45             </label></p>
46
47          <!-- <input type = "password"> inserts -->
48          <!-- textbox whose display is masked   -->
49          <!-- with asterisk characters          -->
50          <p>
51             <label>E-mail Address:
52                <input name = "email" type = "password"
53                   size = "25" />
54             </label>
55          </p>
56
57          <p>
58             <strong>Things you liked:</strong><br />
59
60             <label>Site design
61             <input name = "thingsliked" type = "checkbox"
62                value = "Design" /></label>
63
64             <label>Links
65             <input name = "thingsliked" type = "checkbox"
66                value = "Links" /></label>
67
68             <label>Ease of use
69             <input name = "thingsliked" type = "checkbox"
70                value = "Ease" /></label>
71
72             <label>Images
73             <input name = "thingsliked" type = "checkbox"
74                value = "Images" /></label>
75
76             <label>Source code
```

Fig. E.13 Form with textareas, password boxes and checkboxes. (Part 2 of 3.)

```
77               <input name = "thingsliked" type = "checkbox"
78                  value = "Code" /></label>
79            </p>
80
81            <p>
82               <input type = "submit" value =
83                  "Submit Your Entries" />
84
85               <input type = "reset" value =
86                  "Clear Your Entries" />
87            </p>
88
89         </form>
90
91      </body>
92   </html>
```

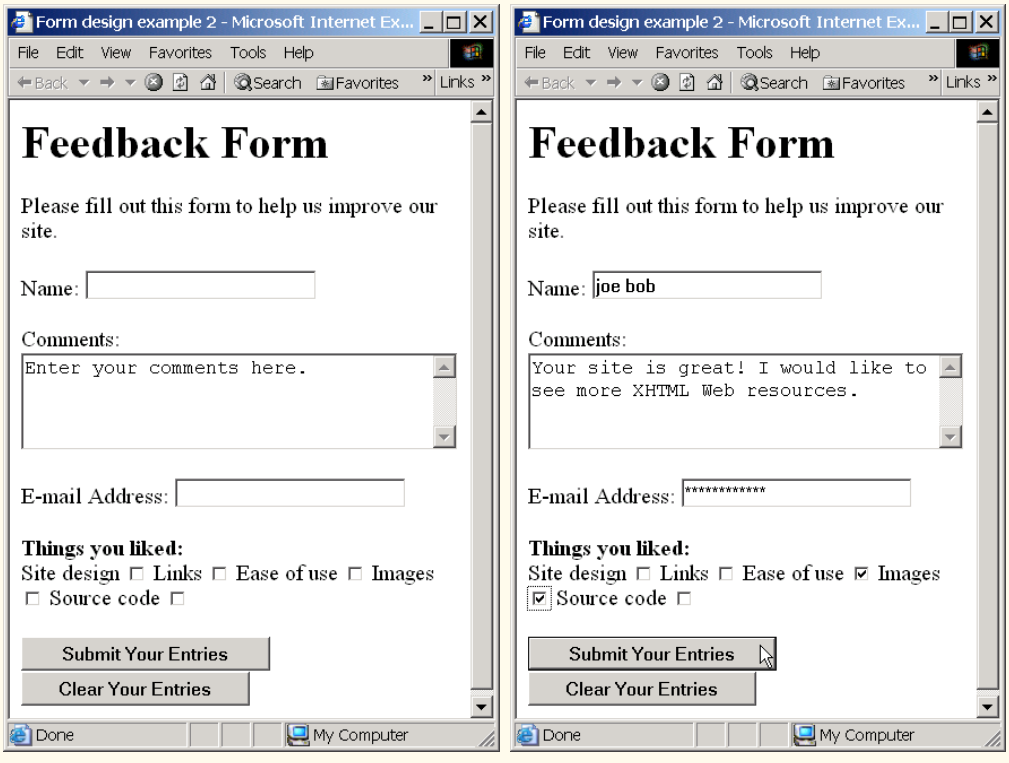


Fig. E.13 Form with textareas, password boxes and checkboxes. (Part 3 of 3.)

Common Programming Error E.7

When your **`form`** *has several checkboxes with the same* **`name`***, you must make sure that they have different* **`value`***s, or the scripts running on the Web server will not be able to distinguish between them.*

Common Programming Error E.8

When using a group of radio buttons in a form, forgetting to set the **`name`** *attributes to the same name is a logic error that lets the user select all of the radio buttons at the same time.*

The **`select`** element (lines 123–136) provides a drop-down list from which the user can select an item. The **`name`** attribute identifies the drop-down list. The **`option`** element (lines 124–135) adds items to the drop-down list. The **`option`** element's **`selected`** *attribute* specifies which item initially is displayed as the selected item in the **`select`** element.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. E.14: form3.html  -->
6  <!-- Form design example 3. -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Form design example 3</title>
11    </head>
12
13    <body>
14
15       <h1>Feedback Form</h1>
16
17       <p>Please fill out this form to help
18          us improve our site.</p>
19
20       <form method = "post" action = "/cgi-bin/formmail">
21
22          <p>
23             <input type = "hidden" name = "recipient"
24                value = "deitel@deitel.com" />
25
26             <input type = "hidden" name = "subject"
27                value = "Feedback Form" />
28
29             <input type = "hidden" name = "redirect"
30                value = "main.html" />
31          </p>
32
33          <p>
34             <label>Name:
35                <input name = "name" type = "text" size = "25" />
36             </label>
37          </p>
38
39          <p>
40             <label>Comments:<br />
```

Fig. E.14 Form including radio buttons and drop-down lists. (Part 1 of 4.)

```
41                <textarea name = "comments" rows = "4"
42                   cols = "36"></textarea>
43             </label>
44          </p>
45
46          <p>
47             <label>E-mail Address:
48                <input name = "email" type = "password"
49                   size = "25" />
50             </label>
51          </p>
52
53          <p>
54             <strong>Things you liked:</strong><br />
55
56             <label>Site design
57                <input name = "thingsliked" type = "checkbox"
58                   value = "Design" />
59             </label>
60
61             <label>Links
62                <input name = "thingsliked" type = "checkbox"
63                   value = "Links" />
64             </label>
65
66             <label>Ease of use
67                <input name = "thingsliked" type = "checkbox"
68                   value = "Ease" />
69             </label>
70
71             <label>Images
72                <input name = "thingsliked" type = "checkbox"
73                   value = "Images" />
74             </label>
75
76             <label>Source code
77                <input name = "thingsliked" type = "checkbox"
78                   value = "Code" />
79             </label>
80
81          </p>
82
83          <!-- <input type = "radio" /> creates one radio   -->
84          <!-- button. The difference between radio buttons -->
85          <!-- and checkboxes is that only one radio button -->
86          <!-- in a group can be selected.                  -->
87          <p>
88             <strong>How did you get to our site?:</strong><br />
89
90             <label>Search engine
91                <input name = "howtosite" type = "radio"
92                   value = "search engine" checked = "checked" />
93             </label>
```

Fig. E.14 Form including radio buttons and drop-down lists. (Part 2 of 4.)

```
94
95             <label>Links from another site
96                <input name = "howtosite" type = "radio"
97                   value = "link" />
98             </label>
99
100            <label>Deitel.com Web site
101               <input name = "howtosite" type = "radio"
102                  value = "deitel.com" />
103            </label>
104
105            <label>Reference in a book
106               <input name = "howtosite" type = "radio"
107                  value = "book" />
108            </label>
109
110            <label>Other
111               <input name = "howtosite" type = "radio"
112                  value = "other" />
113            </label>
114
115         </p>
116
117         <p>
118            <label>Rate our site:
119
120               <!-- <select> tag presents a drop-down -->
121               <!-- list with choices indicated by    -->
122               <!-- <option> tags                     -->
123               <select name = "rating">
124                  <option selected = "selected">Amazing</option>
125                  <option>10</option>
126                  <option>9</option>
127                  <option>8</option>
128                  <option>7</option>
129                  <option>6</option>
130                  <option>5</option>
131                  <option>4</option>
132                  <option>3</option>
133                  <option>2</option>
134                  <option>1</option>
135                  <option>Awful</option>
136               </select>
137
138            </label>
139         </p>
140
141         <p>
142            <input type = "submit" value =
143               "Submit Your Entries" />
144
145            <input type = "reset" value = "Clear Your Entries" />
146         </p>
```

Fig. E.14 Form including radio buttons and drop-down lists. (Part 3 of 4.)

```
147
148         </form>
149
150      </body>
151   </html>
```

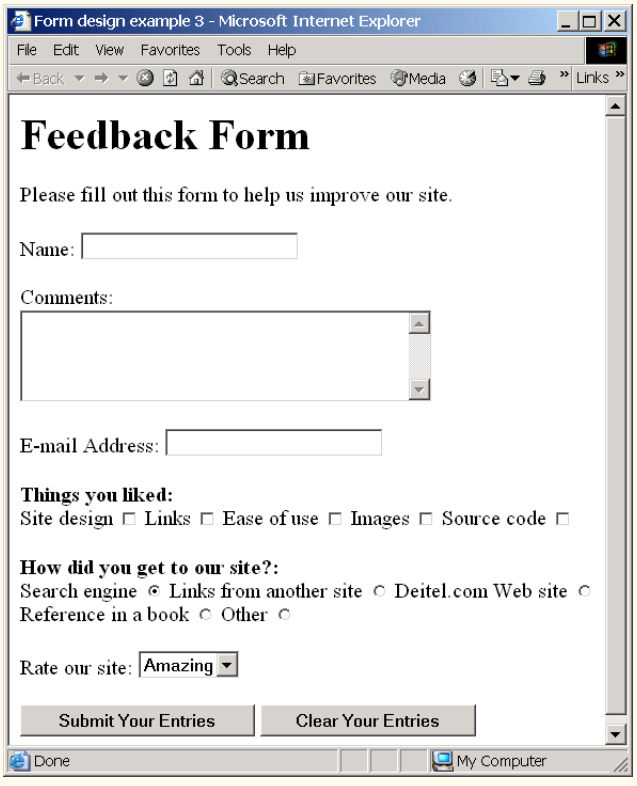


Fig. E.14 Form including radio buttons and drop-down lists. (Part 4 of 4.)

E.14 Internet and World Wide Web Resources

www.w3.org/TR/xhtml1
The *XHTML 1.0 Recommendation* contains general information, information on compatibility issues, document type definition information, definitions, terminology and much more relating to XHTML.

www.xhtml.org
XHTML.org provides XHTML development news and links to other XHTML resources, which include books and articles.

www.w3schools.com/xhtml/default.asp
The *XHTML School* provides XHTML quizzes and references. This page also contains links to XHTML syntax, validation and document type definitions.

hotwired.lycos.com/webmonkey/00/50/index2a.html
This site provides an article about XHTML. Key sections of the article overview XHTML and discuss tags, attributes and anchors.

wdvl.com/Authoring/Languages/XML/XHTML
The Web Developers' Virtual Library provides an introduction to XHTML. This site also contains articles, examples and links to other technologies.

SUMMARY

- XHTML (Extensible Hypertext Markup Language) is a markup language for creating Web pages.
- A key issue when using XHTML is the separation of the presentation of a document (i.e., the document's appearance when rendered by a browser) from the structure of the information in the document.
- In XHTML, text is marked up with elements, delimited by tags that are names contained in pairs of angle brackets. Some elements may contain additional markup called attributes, which provide additional information about the element.
- A machine that runs specialized piece of software called a Web server stores XHTML documents.
- XHTML documents that are syntactically correct are guaranteed to render properly. XHTML documents that contain syntax errors might not display properly.
- Every XHTML document contains a start **<html>** tag and an end **</html>** tag.
- Comments in XHTML always begin with **<!--** and end with **-->**. The browser ignores all text inside a comment.
- Every XHTML document contains a **head** element, which generally contains information, such as a title, and a **body** element, which contains the page content. Information in the **head** element generally is not rendered in the display window but could be made available to the user through other means.
- The **title** element names a Web page. The title usually appears in the colored bar (called the title bar) at the top of the browser window and also appears as the text identifying a page when users add your page to their list of **Favorites** or **Bookmarks**.
- The body of an XHTML document is the area in which the document's content is placed. The content may include text and tags.
- All text placed between the ** ** and ** ** tags form one paragraph.
- XHTML provides six headers (**h1** through **h6**) for specifying the relative importance of information. Header element **h1** is considered the most significant header and is rendered in a larger font than the other five headers. Each successive header element (i.e., **h2**, **h3**, etc.) is rendered in a smaller font.
- Web browsers typically underline text hyperlinks and color them blue by default.
- The **** tag usually causes a browser to render text in a bold font.
- Users can insert links with the **a** (anchor) element. The most important attribute for the **a** element is **href**, which specifies the resource (e.g., page, file, e-mail address) being linked.
- Anchors can link to an e-mail address using a **mailto** URL. When someone clicks this type of anchored link, most browsers launch the default e-mail program (e.g., Outlook Express) to initiate e-mail messages to the linked addresses.
- The **img** element's **src** attribute specifies an image's location. Optional attributes **width** and **height** specify the image width and height, respectively. Images are measured in pixels ("picture elements"), which represent dots of color on the screen.
- The **alt** attribute makes Web pages more accessible to users with disabilities, especially those with vision impairments.

- Some XHTML elements are empty elements, contain only attributes and do not mark up text. Empty elements (e.g., **`img`**) must be terminated, either by using the forward slash character (**`/`**) or by explicitly writing an end tag.
- The **`br`** element causes most browsers to render a line break. Any markup or text following a **`br`** element is rendered on the next line.
- XHTML provides special characters or entity references (in the form **`&`**code**`;`**) for representing characters that cannot be marked up.
- Most browsers render a horizontal rule, indicated by the **`<hr />`** tag, as a horizontal line. The **`hr`** element also inserts a line break above and below the horizontal line.
- The unordered list element **`ul`** creates a list in which each item in the list begins with a bullet symbol (called a disc). Each entry in an unordered list is an **`li`** (list item) element. Most Web browsers render these elements with a line break and a bullet symbol at the beginning of the line.
- Lists may be nested to represent hierarchical data relationships.
- Attribute **`type`** specifies the sequence type (i.e., the set of numbers or letters used in the ordered list).
- XHTML tables mark up tabular data and are one of the most frequently used features in XHTML.
- The **`table`** element defines an XHTML table. Attribute **`border`** specifies the table's border width, in pixels. Tables without borders set this attribute to **`"0"`**.
- Element **`summary`** summarizes the table's contents and is used by speech devices to make the table more accessible to users with visual impairments.
- Element **`caption`** describe's the table's content. The text inside the **`<caption>`** tag is rendered above the table in most browsers.
- A table can be split into three distinct sections: head (**`thead`**), body (**`tbody`**) and foot (**`tfoot`**). The head section contains information such as table titles and column headers. The table body contains the primary table data. The table foot contains information such as footnotes.
- Element **`tr`**, or table row, defines individual table rows. Element **`th`** defines a header cell. Text in **`th`** elements usually is centered and displayed in bold by most browsers. This element can be present in any section of the table.
- Data within a row are defined with **`td`**, or table data, elements.
- Element **`colgroup`** groups and formats columns. Each **`col`** element can format any number of columns (specified with the **`span`** attribute).
- The document author has the ability to merge data cells with the **`rowspan`** and **`colspan`** attributes. The values assigned to these attributes specify the number of rows or columns occupied by the cell. These attributes can be placed inside any data-cell tag.
- XHTML provides forms for collecting information from users. Forms contain visual components, such as buttons that users click. Forms may also contain non-visual components, called hidden inputs, which are used to store any data, such as e-mail addresses and XHTML document file names used for linking.
- A form begins with the **`form`** element. Attribute **`method`** specifies how the form's data is sent to the Web server.
- The **`"text"`** input inserts a text box into the form. Text boxes allow the user to input data.
- The **`input`** element's **`size`** attribute specifies the number of characters visible in the **`input`** element. Optional attribute **`maxlength`** limits the number of characters input into a text box.
- The **`"submit"`** input submits the data entered in the form to the Web server for processing. Most Web browsers create a button that submits the form data when clicked. The **`"reset"`** input allows a user to reset all **`form`** elements to their default values.

- The **textarea** element inserts a multiline text box, called a text area, into a form. The number of rows in the text area is specified with the **rows** attribute and the number of columns (i.e., characters) is specified with the **cols** attribute.
- The **"password"** input inserts a password box into a form. A password box allows users to enter sensitive information, such as credit-card numbers and passwords, by "masking" the information input with another character. Asterisks are the masking character used for password boxes. The actual value input is sent to the Web server, not the asterisks that mask the input.
- The checkbox input allows the user to make a selection. When the checkbox is selected, a check mark appears in the checkbox. Otherwise, the checkbox is empty. Checkboxes can be used individually and in groups. Checkboxes that are part of the same group have the same **name**.
- A radio button is similar in function and use to a checkbox, except that only one radio button in a group can be selected at any time. All radio buttons in a group have the same **name** attribute value and have different attribute **value**s.
- The **select** input provides a drop-down list of items. The **name** attribute identifies the drop-down list. The **option** element adds items to the drop-down list. The **selected** attribute, like the **checked** attribute for radio buttons and checkboxes, specifies which list item is displayed initially.

TERMINOLOGY

<!--…--> (XHTML comment)
a element (**<a>…**)
action attribute
alt attribute
& (**&** special character)
anchor
angle brackets (**< >**)
attribute
body element
border attribute
br (line break) element
browser request
<caption> tag
checkbox
checked attribute
col element
colgroup element
cols attribute
colspan attribute
comments in XHTML
© (© special character)
disc
element
e-mail anchor
empty tag
form
form element
head element
header
header cell
header elements (**h1** through **h6**)
height attribute
hexadecimal code
hidden input element
<hr /> tag (horizontal rule)
href attribute
.htm (XHTML file-name extension)
.html (XHTML file-name extension)
<html> tag
hyperlink
image hyperlink
img element
input element
level of nesting
<li> (list item) tag
linked document
mailto: URL
markup language
maxlength attribute
method attribute
name attribute
nested list
nested tag
ol (ordered list) element
p (paragraph) element
password box
"radio" (attribute value)
rows attribute (**textarea**)
rowspan attribute (**tr**)
selected attribute
size attribute (**input**)
special character