5

Introduction to XHTML: Part 2

Objectives

- To be able to create tables with rows and columns of data.
- To be able to control table formatting.
- To be able to create and use forms.
- To be able to create and use image maps to aid in Web-page navigation.
- To be able to make Web pages accessible to search engines using **<meta>** tags.
- To be able to use the **frameset** element to display multiple Web pages in a single browser window.

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

Outline

5.1 Introduction

In the previous chapter, we introduced XHTML. We built several complete Web pages featuring text, hyperlinks, images, horizontal rules and line breaks. In this chapter, we discuss more substantial XHTML features, including presentation of information in *tables* and *incorporating forms* for collecting information from a Web-page visitor. We also introduce *internal linking* and *image maps* for enhancing Web page navigation and *frames* for displaying multiple documents in the browser.

By the end of this chapter, you will be familiar with the most commonly used XHTML features and will be able to create more complex Web documents. In Chapter 6, we discuss how to make Web pages more visually appealing by manipulating fonts, colors and text.

5.2 Basic XHTML Tables

This section presents XHTML *tables*—a frequently used feature that organizes data into rows and columns. Our first example (Fig. 5.1) 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 Strict//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4
5  <!-- Fig. 5.1: table1.html    -->
6  <!-- Creating a basic table  -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>A simple XHTML table</title>
11    </head>
```

Fig. 5.1 XHTML table (part 1 of 3).

```
12
13       <body>
14
15          <!-- the <table> tag opens a table -->
16          <table border = "1" width = "40%"
17             summary = "This table provides information about
18                the price of fruit">
19
20             <!-- the <caption> tag summarizes the table's    -->
21             <!-- contents (this helps the visually impaired) -->
22             <caption><strong>Price of Fruit</strong></caption>
23
24             <!-- the <thead> is the first section of a table -->
25             <!-- it formats the table header area            -->
26             <thead>
27                <tr>              <!-- <tr> inserts a table row -->
28                   <th>Fruit</th> <!-- insert a heading cell -->
29                   <th>Price</th>
30                </tr>
31             </thead>
32
33             <!-- all table content is enclosed  -->
34             <!-- within the <tbody>             -->
35             <tbody>
36                <tr>
37                   <td>Apple</td> <!-- insert a data cell -->
38                   <td>$0.25</td>
39                </tr>
40
41                <tr>
42                   <td>Orange</td>
43                   <td>$0.50</td>
44                </tr>
45
46                <tr>
47                   <td>Banana</td>
48                   <td>$1.00</td>
49                </tr>
50
51                <tr>
52                   <td>Pineapple</td>
53                   <td>$2.00</td>
54                </tr>
55             </tbody>
56
57             <!-- the <tfoot> is the last section of a table -->
58             <!-- it formats the table footer                -->
59             <tfoot>
60                <tr>
61                   <th>Total</th>
62                   <th>$3.75</th>
63                </tr>
64             </tfoot>
```

Fig. 5.1 XHTML table (part 2 of 3).

```
65
66          </table>
67
68       </body>
69    </html>
```

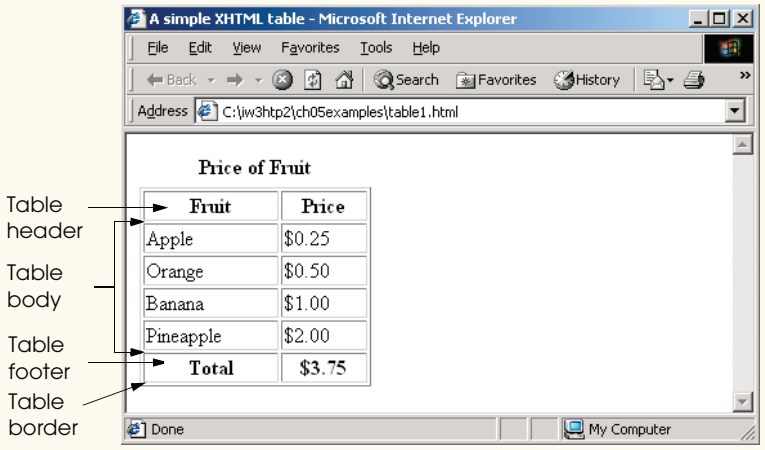

Fig. 5.1 XHTML table (part 3 of 3).

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.

Testing and Debugging Tip 5.1

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

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. We discuss accessibility programming in detail in Chapter 34, Accessibility.

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 and display text formatted by **`th`** (table header column) elements 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 35–55) 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 59–64) is defined with a **tfoot** (table foot) element and represents a footer. Common text 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.

5.3 Intermediate XHTML Tables and Formatting

In the previous section, we explored the structure of a basic table. In Fig. 5.2, we enhance our discussion of tables by introducing elements and attributes that allow the document author to build more complex tables.

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.

```
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. 5.2: table2.html      -->
6  <!-- Intermediate table design -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Internet and WWW How to Program - Tables</title>
11    </head>
12
13    <body>
14
15       <h1>Table Example Page</h1>
16
17       <table border = "1">
18          <caption>Here is a more complex sample table.</caption>
19
20          <!-- <colgroup> and <col> tags are used to -->
21          <!-- format entire columns                 -->
22          <colgroup>
23
24             <!-- span attribute determines how many columns -->
25             <!-- the <col> tag affects                      -->
26             <col align = "right" span = "1" />
```

Fig. 5.2 Complex XHTML table (part 1 of 3).

```
27          </colgroup>
28
29          <thead>
30
31             <!-- rowspans and colspans merge the specified    -->
32             <!-- number of cells vertically or horizontally   -->
33             <tr>
34
35                <!-- merge two rows -->
36                <th rowspan = "2">
37                   <img src = "camel.gif" width = "205"
38                      height = "167" alt = "Picture of a camel" />
39                </th>
40
41                <!-- merge four columns -->
42                <th colspan = "4" valign = "top">
43                   <h1>Camelid comparison</h1><br />
44                   <p>Approximate as of 9/2002</p>
45                </th>
46             </tr>
47
48             <tr valign = "bottom">
49                <th># of Humps</th>
50                <th>Indigenous region</th>
51                <th>Spits?</th>
52                <th>Produces Wool?</th>
53             </tr>
54
55          </thead>
56
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. 5.2 Complex XHTML table (part 2 of 3).

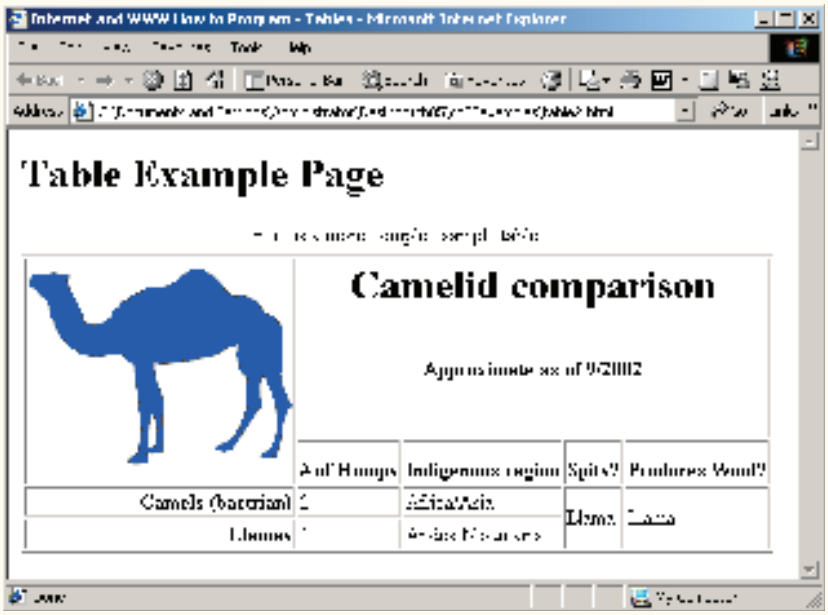

Fig. 5.2 Complex XHTML table (part 3 of 3).

Common Programming Error 5.1

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 may inadvertently create more columns and rows than you originally intended.*

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).

5.4 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, images, etc.). These resources are either located 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 5.3 sends the form data to the Web server which passes the form data to a *CGI* (*Common Gateway Interface*) script (i.e., a program) written in Perl, C or some other language. The script 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. We discuss Web servers in Chapter 21. [*Note*: This example demonstrates client-side functionality. If the form is submitted (by clicking **Submit Your Entries**) an error occurs. In later chapters such as Perl and Python, we present the server-side programming necessary to process information entered into a form.]

```
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. 5.3: form.html    -->
6  <!-- Form Design Example 1 -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Internet and WWW How to Program - Forms</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       <!-- this tag starts the form, gives the    -->
21       <!-- method of sending information and the -->
22       <!-- location of form scripts               -->
23       <form method = "post" action = "/cgi-bin/formmail">
24
25          <p>
26             <!-- hidden inputs contain non-visual -->
27             <!-- information                      -->
28             <input type = "hidden" name = "recipient"
29                value = "deitel@deitel.com" />
30             <input type = "hidden" name = "subject"
31                value = "Feedback Form" />
32             <input type = "hidden" name = "redirect"
33                value = "main.html" />
34          </p>
35
36          <!-- <input type = "text"> inserts a text box -->
37          <p><label>Name:
38                <input name = "name" type = "text" size = "25"
39                   maxlength = "30" />
40             </label></p>
```

Fig. 5.3 Simple form with hidden fields and a text box (part 1 of 2).

```
41
42          <p>
43             <!-- input types "submit" and "reset" insert  -->
44             <!-- buttons for submitting and clearing the  -->
45             <!-- form's contents                          -->
46             <input type = "submit" value =
47                "Submit Your Entries" />
48             <input type = "reset" value =
49                "Clear Your Entries" />
50          </p>
51
52       </form>
53
54 </html>
```

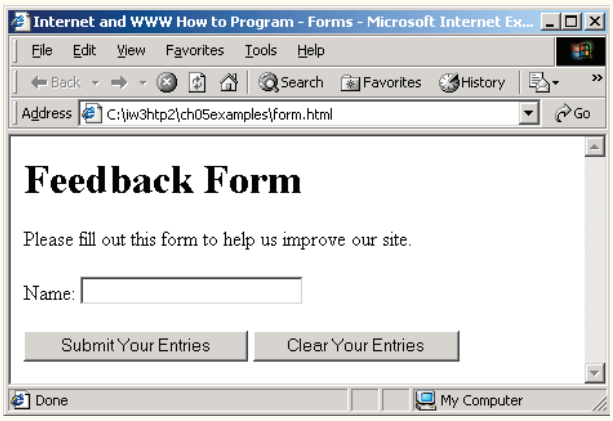

Fig. 5.3 Simple form with hidden fields and a text box (part 2 of 2).

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 28–33 define three **input** elements that specify data to provide to the script that processes the form (also called the *form handler*). These three **input** element 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 5.1

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 5.2

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 46–49. 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**).

5.5 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 5.4 contains a form that solicits user feedback about a Web site.

The **textarea** element (lines 37–39) inserts a multiline text box, called a *text area*, 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.

```
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. 5.4 Form with textareas, password boxes and checkboxes (part 1 of 4).

```
<!-- Fig. 5.4: form2.html  -->
<!-- Form Design Example 2 -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Internet and WWW How to Program - Forms</title>
   </head>

   <body>

      <h1>Feedback Form</h1>

      <p>Please fill out this form to help
         us improve our site.</p>

      <form method = "post" action = "/cgi-bin/formmail">

         <p>
            <input type = "hidden" name = "recipient"
               value = "deitel@deitel.com" />
            <input type = "hidden" name = "subject"
               value = "Feedback Form" />
            <input type = "hidden" name = "redirect"
               value = "main.html" />
         </p>

         <p><label>Name:
            <input name = "name" type = "text" size = "25" />
         </label></p>

         <!-- <textarea> creates a multiline textbox -->
         <p><label>Comments:<br />
            <textarea name = "comments" rows = "4" cols = "36">
Enter your comments here.
            </textarea>
         </label></p>

         <!-- <input type = "password"> inserts a    -->
         <!-- textbox whose display is masked with  -->
         <!-- asterisk characters                   -->
         <p><label>E-mail Address:
            <input name = "email" type = "password"
               size = "25" />
         </label></p>

         <p>
            <strong>Things you liked:</strong><br />

            <label>Site design
            <input name = "thingsliked" type = "checkbox"
               value = "Design" /></label>

```

Fig. 5.4 Form with textareas, password boxes and checkboxes (part 2 of 4).

```
57                <label>Links
58                <input name = "thingsliked" type = "checkbox"
59                   value = "Links" /></label>
60
61                <label>Ease of use
62                <input name = "thingsliked" type = "checkbox"
63                   value = "Ease" /></label>
64
65                <label>Images
66                <input name = "thingsliked" type = "checkbox"
67                   value = "Images" /></label>
68
69                <label>Source code
70                <input name = "thingsliked" type = "checkbox"
71                   value = "Code" /></label>
72             </p>
73
74             <p>
75                <input type = "submit" value =
76                   "Submit Your Entries" />
77                <input type = "reset" value =
78                   "Clear Your Entries" />
79             </p>
80
81          </form>
82    </html>
```

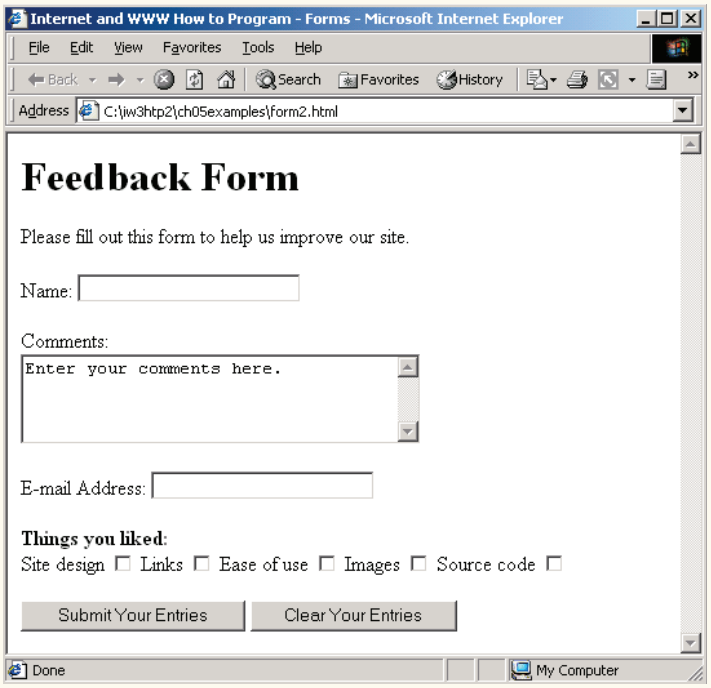

Fig. 5.4 Form with textareas, password boxes and checkboxes (part 3 of 4).

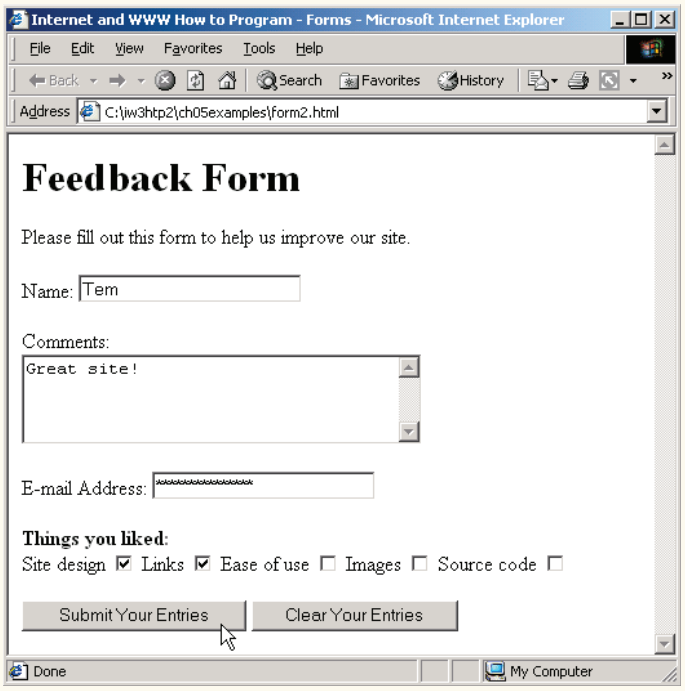

Fig. 5.4 Form with textareas, password boxes and checkboxes (part 4 of 4).

The **"password"** input in lines 46–47, 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 54–71 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"**).

Common Programming Error 5.3

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.*

We continue our discussion of forms by presenting a third example that introduces several more form elements from which users can make selections (Fig. 5.5). In this example, we introduce two new **input** types. The first type is the *radio button* (lines 76–94) 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** attributes and are distinguished by their different **value** attributes. The attribute-value pair **checked = "checked"** (line 77) indicates which radio button, if any, is selected initially. The **checked** attribute also applies to checkboxes.

```
<?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. 5.5: form3.html  -->
<!-- Form Design Example 3 -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Internet and WWW How to Program - Forms</title>
   </head>

   <body>

      <h1>Feedback Form</h1>

      <p>Please fill out this form to help
         us improve our site.</p>

      <form method = "post" action = "/cgi-bin/formmail">

         <p>
            <input type = "hidden" name = "recipient"
               value = "deitel@deitel.com" />
            <input type = "hidden" name = "subject"
               value = "Feedback Form" />
            <input type = "hidden" name = "redirect"
               value = "main.html" />
         </p>

         <p><label>Name:
               <input name = "name" type = "text" size = "25" />
            </label></p>

         <p><label>Comments:<br />
               <textarea name = "comments" rows = "4"
                  cols = "36"></textarea>
            </label></p>

         <p><label>E-mail Address:
               <input name = "email" type = "password"
                  size = "25" /></label></p>

         <p>
            <strong>Things you liked:</strong><br />

            <label>Site design
               <input name = "thingsliked" type = "checkbox"
                  value = "Design" /></label>

            <label>Links
               <input name = "thingsliked" type = "checkbox"
                  value = "Links" /></label>
```

Fig. 5.5 Form including radio buttons and drop-down lists (part 1 of 4).

```
54
55             <label>Ease of use
56                <input name = "thingsliked" type = "checkbox"
57                   value = "Ease" /></label>
58
59             <label>Images
60                <input name = "thingsliked" type = "checkbox"
61                   value = "Images" /></label>
62
63             <label>Source code
64                <input name = "thingsliked" type = "checkbox"
65                   value = "Code" /></label>
66          </p>
67
68          <!-- <input type = "radio" /> creates a radio     -->
69          <!-- button. The difference between radio buttons -->
70          <!-- and checkboxes is that only one radio button -->
71          <!-- in a group can be selected.                  -->
72          <p>
73             <strong>How did you get to our site?:</strong><br />
74
75             <label>Search engine
76                <input name = "howtosite" type = "radio"
77                   value = "search engine" checked = "checked" />
78             </label>
79
80             <label>Links from another site
81                <input name = "howtosite" type = "radio"
82                   value = "link" /></label>
83
84             <label>Deitel.com Web site
85                <input name = "howtosite" type = "radio"
86                   value = "deitel.com" /></label>
87
88             <label>Reference in a book
89                <input name = "howtosite" type = "radio"
90                   value = "book" /></label>
91
92             <label>Other
93                <input name = "howtosite" type = "radio"
94                   value = "other" /></label>
95
96          </p>
97
98          <p>
99             <label>Rate our site:
100
101               <!-- the <select> tag presents a drop-down -->
102               <!-- list with choices indicated by the    -->
103               <!-- <option> tags                         -->
104               <select name = "rating">
105                  <option selected = "selected">Amazing</option>
106                  <option>10</option>
```

Fig. 5.5 Form including radio buttons and drop-down lists (part 2 of 4).

```
107                   <option>9</option>
108                   <option>8</option>
109                   <option>7</option>
110                   <option>6</option>
111                   <option>5</option>
112                   <option>4</option>
113                   <option>3</option>
114                   <option>2</option>
115                   <option>1</option>
116                   <option>Awful</option>
117                </select>
118
119             </label>
120          </p>
121
122          <p>
123             <input type = "submit" value =
124                "Submit Your Entries" />
125             <input type = "reset" value = "Clear Your Entries" />
126          </p>
127
128       </form>
129
130    </body>
131 </html>
```

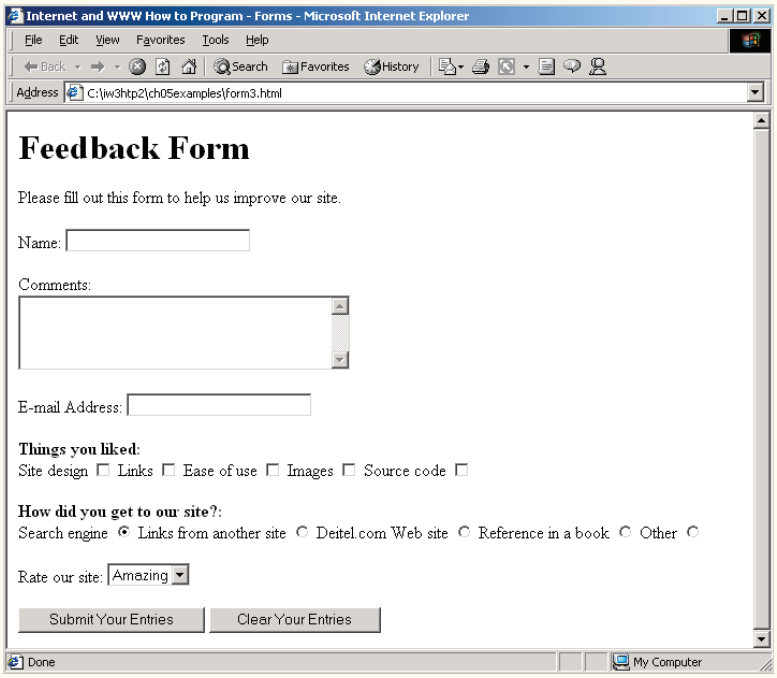

Fig. 5.5 Form including radio buttons and drop-down lists (part 3 of 4).

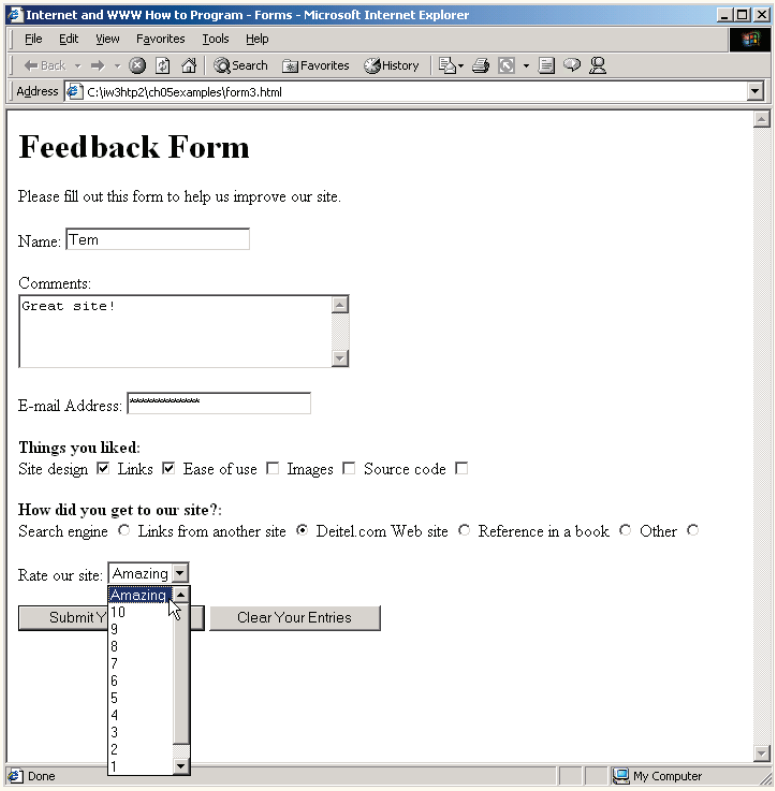

Fig. 5.5 Form including radio buttons and drop-down lists (part 4 of 4).

Common Programming Error 5.4

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

The **`select`** element (lines 104–117) provides a drop-down list of items from which the user can select an item. The **`name`** attribute identifies the drop-down list. The **`option`** element (lines 105–116) 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.

5.6 Internal Linking

In Chapter 4, we discussed how to hyperlink one Web page to another. Figure 5.6 introduces *internal linking*—a mechanism that enables the user to jump between locations in the same document. Internal linking is useful for long documents that contain many sections. Clicking an internal link enables users to find a section without scrolling through the entire document.

```
<?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. 5.6: links.html -->
<!-- Internal Linking     -->

<html xmlns = "http://www.w3.org/1999/xhtml">
   <head>
      <title>Internet and WWW How to Program - List</title>
   </head>

   <body>

      <!-- <a name = ".."></a> creates an internal hyperlink -->
      <p><a name = "features"></a></p>
      <h1>The Best Features of the Internet</h1>

      <!-- an internal link's address is "#linkname"         -->
      <p><a href = "#ceos">Go to <em>Favorite CEOs</em></a></p>

      <ul>
         <li>You can meet people from countries
            around the world.</li>

         <li>You have access to new media as it becomes public:
            <ul>
               <li>New games</li>
               <li>New applications
                  <ul>
                     <li>For Business</li>
                     <li>For Pleasure</li>
                  </ul>
               </li>

               <li>Around the clock news</li>
               <li>Search Engines</li>
               <li>Shopping</li>
               <li>Programming
                  <ul>
                     <li>XHTML</li>
                     <li>Java</li>
                     <li>Dynamic HTML</li>
                     <li>Scripts</li>
                     <li>New languages</li>
                  </ul>
               </li>
            </ul>
         </li>

         <li>Links</li>
         <li>Keeping in touch with old friends</li>
```

Fig. 5.6 Using internal hyperlinks to make pages more navigable (part 1 of 2).

```
53          <li>It is the technology of the future!</li>
54       </ul>
55
56       <!-- named anchor -->
57       <p><a name = "ceos"></a></p>
58       <h1>My 3 Favorite <em>CEOs</em></h1>
59
60       <p>
61
62          <!-- internal hyperlink to features -->
63          <a href = "#features">Go to <em>Favorite Features</em>
64          </a></p>
65
66       <ol>
67          <li>Bill Gates</li>
68          <li>Steve Jobs</li>
69          <li>Michael Dell</li>
70       </ol>
71
72    </body>
73 </html>
```

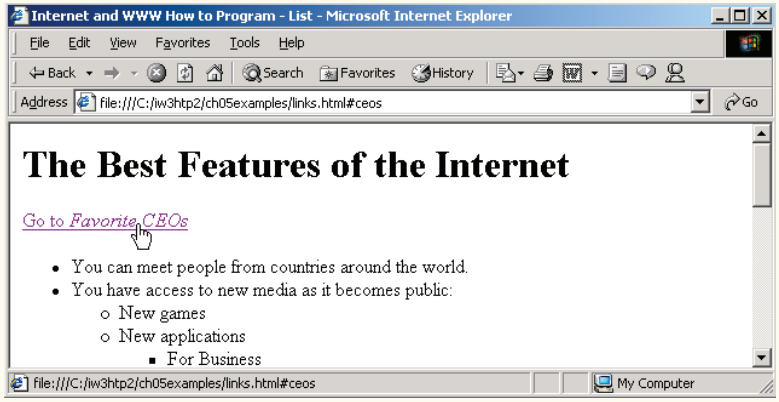

Fig. 5.6 Using internal hyperlinks to make pages more navigable (part 2 of 2).

Line 16 contains a *named anchor* (called **features**) for an internal hyperlink. To link to this type of anchor inside the same Web page, the href attribute of another anchor element includes the named anchor preceded with a pound sign (as in **#features**). Lines 63–64 contain a hyperlink with the anchor **features** as its target. Selecting this hyperlink in a Web browser scrolls the browser window to the **features** anchor at line 16.

Look-and-Feel Observation 5.1

Internal hyperlinks are useful in XHTML documents that contain large amounts of information. Internal links to various sections on the page makes it easier for users to navigate the page. They do not have to scroll to find a specific section.

Although not demonstrated in this example, a hyperlink can specify an internal link in another document by specifying the document name followed by a pound sign and the named anchor as in:

```
href = "page.html#name"
```

For example, to link to a named anchor called **booklist** in **books.html**, **href** is assigned **"books.html#booklist"**.

5.7 Creating and Using Image Maps

In Chapter 4, we demonstrated how images can be used as hyperlinks to link to other resources on the Internet. In this section, we introduce another technique for image linking called *image maps*, which designate certain areas of an image (called *hotspots*) as links. Figure 5.7 introduces image maps and hotspots.

```
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. 5.7: picture.html         -->
6  <!-- Creating and Using Image Maps -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>
11          Internet and WWW How to Program - Image Map
12       </title>
13    </head>
14
15    <body>
16
17       <p>
18
19       <!-- the <map> tag defines an image map -->
20       <map id = "picture">
21
22          <!-- shape = "rect" indicates a rectangular    -->
23          <!-- area, with coordinates for the upper-left -->
24          <!-- and lower-right corners                   -->
```

Fig. 5.7 Image with links anchored to an image map (part 1 of 2).

```
25          <area href = "form.html" shape = "rect"
26             coords = "2,123,54,143"
27             alt = "Go to the feedback form" />
28          <area href = "contact.html" shape = "rect"
29             coords = "126,122,198,143"
30             alt = "Go to the contact page" />
31          <area href = "main.html" shape = "rect"
32             coords = "3,7,61,25" alt = "Go to the homepage" />
33          <area href = "links.html" shape = "rect"
34             coords = "168,5,197,25"
35             alt = "Go to the links page" />
36
37          <!-- value "poly" creates a hotspot in the shape -->
38          <!-- of a polygon, defined by coords             -->
39          <area shape = "poly" alt = "E-mail the Deitels"
40             coords = "162,25,154,39,158,54,169,51,183,39,161,26"
41             href = "mailto:deitel@deitel.com" />
42
43          <!-- shape = "circle" indicates a circular -->
44          <!-- area with the given center and radius -->
45          <area href = "mailto:deitel@deitel.com"
46             shape = "circle" coords = "100,36,33"
47             alt = "E-mail the Deitels" />
48       </map>
49
50       <!-- <img src =... usemap = "#id"> indicates that the -->
51       <!-- specified image map is used with this image      -->
52       <img src = "deitel.gif" width = "200" height = "144"
53          alt = "Deitel logo" usemap = "#picture" />
54       </p>
55    </body>
56 </html>
```

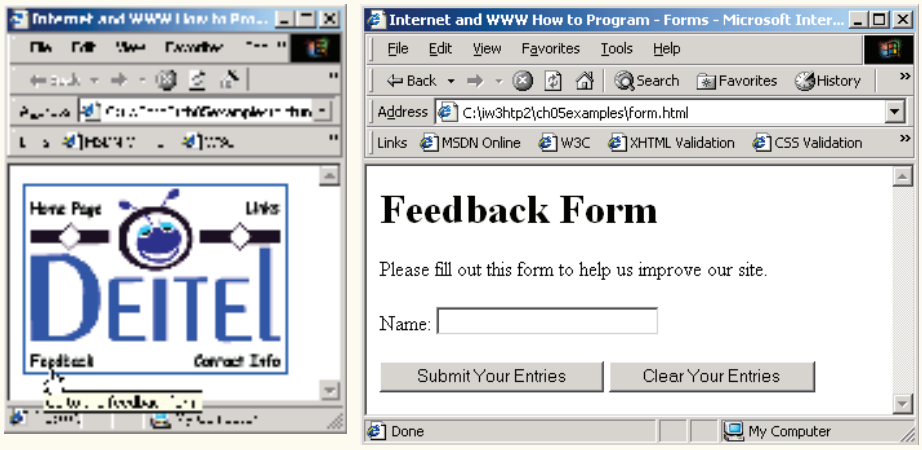

Fig. 5.7 Image with links anchored to an image map (part 2 of 2).

Lines 20–48 define an image maps by using a ***map*** element. Attribute ***id*** (line 20) identifies the image map. If **id** is omitted, the map cannot be referenced by an image. We

discuss how to reference an image map momentarily. Hotspots are defined with **`area`** elements (as shown on lines 25–27). Attribute **`href`** (line 25) specifies the link's target (i.e., the resource to which to link). Attributes **`shape`** (line 25) and **`coords`** (line 26) specify the hotspot's shape and coordinates, respectively. Attribute **`alt`** (line 27) provides alternate text for the link.

Common Programming Error 5.5

Not specifying an **`id`** *attribute for a* **`map`** *element prevents an* **`img`** *element from using the* **`map`***'s* **`area`** *elements to define hotspots.*

The markup on lines 25–27 creates a *rectangular hotspot* (**`shape = "rect"`**) for the *coordinates* specified in the **`coords`** attribute. A coordinate pair consists of two numbers representing the location of a point on the *x*-axis and the *y*-axis, respectively. The *x*-axis extends horizontally and the *y*-axis extends vertically from the upper-left corner of the image. Every point on an image has a unique *x*-*y*-coordinate. For rectangular hotspots, the required coordinates are those of the upper-left and lower-right corners of the rectangle. In this case, the upper-left corner of the rectangle is located at 2 on the *x*-axis and 123 on the *y*-axis, annotated as *(2, 123)*. The lower-right corner of the rectangle is at *(54, 143)*. Coordinates are measured in pixels.

Common Programming Error 5.6

Overlapping coordinates of an image map cause the browser to render the first hotspot it encounters for the area.

The map **`area`** (lines 39–41) assigns the **`shape`** attribute **`"poly"`** to create a hotspot in the shape of a polygon using the coordinates in attribute **`coords`**. These coordinates represent each *vertex*, or corner, of the polygon. The browser connects these points with lines to form the hotspot's area.

The map **`area`** (lines 45–47) assigns the **`shape`** attribute **`"circle"`** to create a *circular hotspot*. In this case, the **`coords`** attribute specifies the circle's center coordinates and the circle's radius, in pixels.

To use an image map with an **`img`** element, the **`img`** element's **`usemap`** attribute is assigned the **`id`** of a **`map`**. Lines 52–53 reference the image map named **`"picture"`**. The image map is located within the same document so internal linking is used.

5.8 `meta` Elements

People use search engines to find useful Web sites. Search engines usually catalog sites by following links from page to page and saving identification and classification information for each page. One way that search engines catalog pages is by reading the content in each page's **`meta`** elements, which specify information about a document.

Two important attributes of the **`meta`** element are **`name`**, which identifies the type of **`meta`** element and **`content`**, which provides the information search engines use to catalog pages. Figure 5.8 introduces the **`meta`** element.

Lines 14–16 demonstrate a **`"keywords"`** **`meta`** element. The **`content`** attribute of such a **`meta`** element provides search engines with a list of words that describe a page. These words are compared with words in search requests. Thus, including **`meta`** elements and their **`content`** information can draw more viewers to your site.

```
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. 5.8: main.html -->
6  <!-- <meta> tag          -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Internet and WWW How to Program - Welcome</title>
11
12       <!-- <meta> tags provide search engines with -->
13       <!-- information used to catalog a site      -->
14       <meta name = "keywords" content = "Web page, design,
15          XHTML, tutorial, personal, help, index, form,
16          contact, feedback, list, links, frame, deitel" />
17
18       <meta name = "description" content = "This Web site will
19          help you learn the basics of XHTML and Web page design
20          through the use of interactive examples and
21          instruction." />
22
23    </head>
24
25    <body>
26
27       <h1>Welcome to Our Web Site!</h1>
28
29       <p>We have designed this site to teach about the wonders
30       of <strong><em>XHTML</em></strong>. <em>XHTML</em> is
31       better equipped than <em>HTML</em> to represent complex
32       data on the Internet. <em>XHTML</em> takes advantage of
33       XML's strict syntax to ensure well-formedness. Soon you
34       will know about many of the great new features of
35       <em>XHTML.</em></p>
36
37      <p>Have Fun With the Site!</p>
38
39    </body>
40 </html>
```

Fig. 5.8 Using **meta** to provide keywords and a description .

Lines 18–21 demonstrate a ***"description"*** **meta** element. The **content** attribute of such a **meta** element provides a three- to four-line description of a site, written in sentence form. Search engines also use this description to catalog your site and sometimes display this information as part of the search results.

Software Engineering Observation 5.1

meta *elements are not visible to users and must be placed inside the* ***head*** *section of your XHTML document. If* ***meta*** *elements are not placed in this section, they will not be read by search engines.*

5.9 `frameset` Element

All of the Web pages we have presented in this book have the ability to link to other pages, but can display only one page at a time. Figure 5.9 uses *frames*, which allow the browser to display more than one XHTML document simultaneously, to display the documents in Fig. 5.8 and Fig. 5.10.

Most of our prior examples adhered to the strict XHTML document type. This particular example uses the *frameset* document type—a special XHTML document type specifically for framesets. This new document type is specified in lines 2–3 and is required for documents that define framesets.

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
4
5  <!-- Fig. 5.9: index.html -->
6  <!-- XHTML Frames I       -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Internet and WWW How to Program - Main</title>
11       <meta name = "keywords" content = "Webpage, design,
12          XHTML, tutorial, personal, help, index, form,
13          contact, feedback, list, links, frame, deitel" />
14
15       <meta name = "description" content = "This Web site will
16          help you learn the basics of XHTML and Web page design
17          through the use of interactive examples
18          and instruction." />
19
20    </head>
21
22    <!-- the <frameset> tag sets the frame dimensions     -->
23    <frameset cols = "110,*">
24
25       <!-- frame elements specify which pages -->
26       <!-- are loaded into a given frame      -->
27       <frame name = "leftframe" src = "nav.html" />
28       <frame name = "main" src = "main.html" />
29
30       <noframes>
31          <p>This page uses frames, but your browser does not
32          support them.</p>
33
34          <p>Please, <a href = "nav.html">follow this link to
35          browse our site without frames</a>.</p>
36       </noframes>
37
38    </frameset>
39 </html>
```

Fig. 5.9 Web document containing two frames—navigation and content (part 1 of 2).

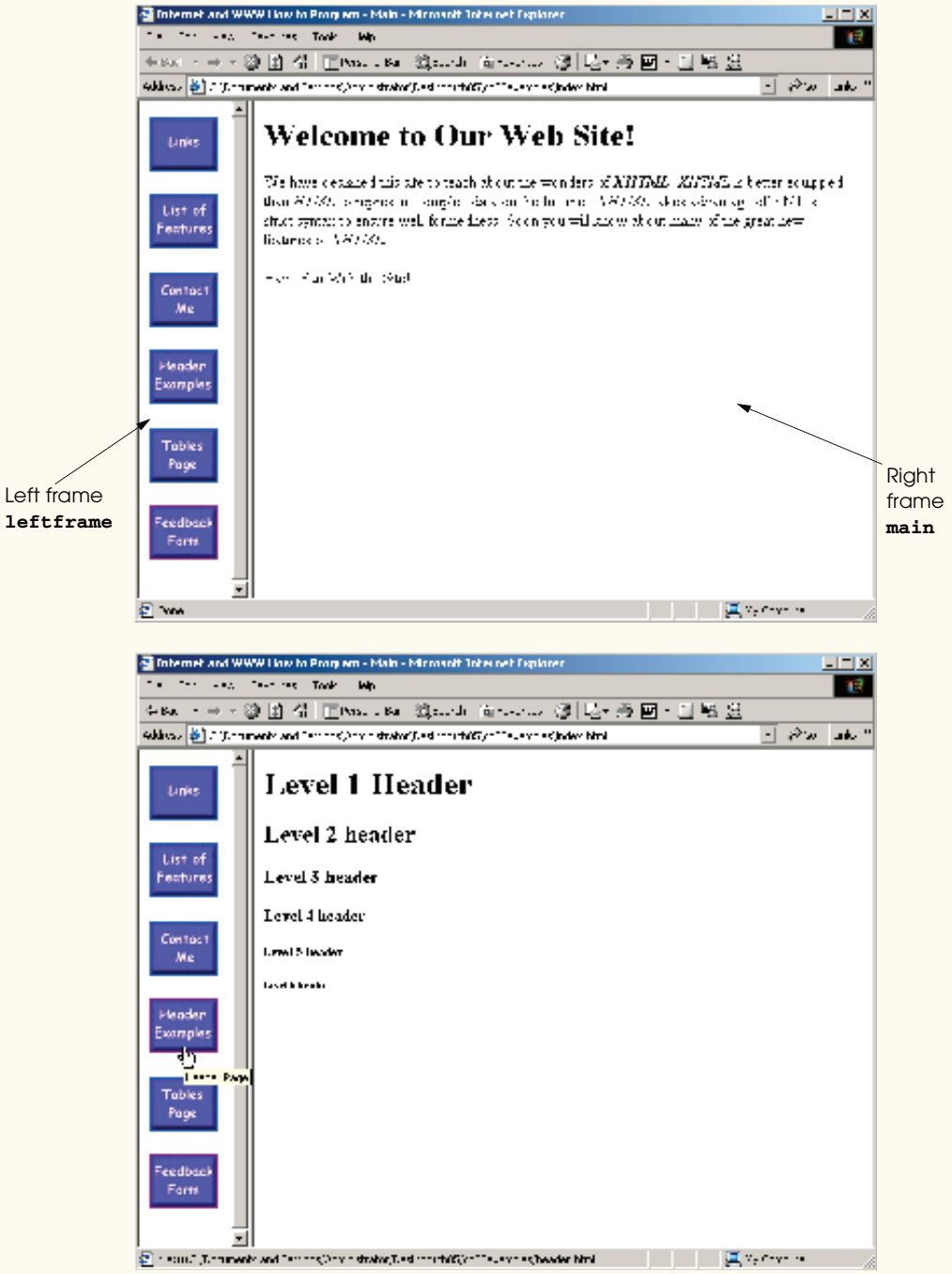

Fig. 5.9 Web document containing two frames—navigation and content (part 2 of 2).

A document that defines a frameset normally consists of an **html** element that contains a **head** element and a ***frameset*** element. The ***<frameset>*** tag (line 23) informs the browser that the page contains frames. Attribute ***cols*** specifies the frameset's column layout. The value of **cols** gives the width of each frame, either in pixels or as a percentage of the browser width. In this case, the attribute **cols = "110,*"** informs the browser that there are two vertical frames. The first frame extends **110** pixels from the left edge of the browser window and the second frame fills the remainder of the browser width (as indicated by the asterisk). Similarly, **frameset** attribute ***rows*** can be used to specify the number of rows and the size of each row in a frameset.

The documents that will be loaded into the **frameset** are specified with ***frame*** elements (lines 27–28 in this example). Attribute **src** specifies the URL of the page to display in the frame. Each frame has **name** and **src** attributes. The first frame (which covers **110** pixels on the left side of the **frameset**) is named **leftframe** and displays the page **nav.html** (Fig. 5.10). The second frame is named main and displays the page **main.html**.

Attribute **name** identifies a frame, enabling hyperlinks in a **frameset** to specify the ***target*** **frame** in which a linked document should display when the user clicks the link. For example

```
<a href = "links.html" target = "main">
```

loads **links.html** in the frame whose **name** is **"main"**.

Not all browsers support frames. XHTML provides the ***noframes*** element (lines 30–36) to enable XHTML document designers to specify alternate content for browsers that do not support frames.

Portability Tip 5.1

*Some browsers do not support frames. Use the **noframes** element inside a **frameset** to direct users to a nonframed version of your site.*

Fig. 5.10 is the Web page displayed in the left frame of Fig. 5.9. This XHTML document provides the navigation buttons that, when clicked, determine which document is displayed in the right frame.

```
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. 5.10: nav.html           -->
6  <!-- Using images as link anchors -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9
10    <head>
11       <title>Internet and WWW How to Program - Navigation Bar
12       </title>
13    </head>
```

Fig. 5.10 XHTML document displayed in the left frame of Fig. 5.9 (part 1 of 2).

```
14
15      <body>
16
17         <p>
18            <a href = "links.html" target = "main">
19               <img src = "buttons/links.jpg" width = "65"
20                  height = "50" alt = "Links Page" />
21            </a><br />
22
23            <a href = "list.html" target = "main">
24               <img src = "buttons/list.jpg" width = "65"
25                  height = "50" alt = "List Example Page" />
26            </a><br />
27
28            <a href = "contact.html" target = "main">
29               <img src = "buttons/contact.jpg" width = "65"
30                  height = "50" alt = "Contact Page" />
31            </a><br />
32
33            <a href = "header.html" target = "main">
34               <img src = "buttons/header.jpg" width = "65"
35                  height = "50" alt = "Header Page" />
36            </a><br />
37
38            <a href = "table1.html" target = "main">
39               <img src = "buttons/table.jpg" width = "65"
40                  height = "50" alt = "Table Page" />
41            </a><br />
42
43            <a href = "form.html" target = "main">
44               <img src = "buttons/form.jpg" width = "65"
45                  height = "50" alt = "Feedback Form" />
46            </a><br />
47         </p>
48
49      </body>
50   </html>
```

Fig. 5.10 XHTML document displayed in the left frame of Fig. 5.9 (part 2 of 2).

Line 27 (Fig. 5.9) displays the XHTML page in Fig. 5.10. Anchor attribute **`target`** (line 18 in Fig. 5.10) specifies that the linked documents are loaded in frame **`main`** (line 28 in Fig. 5.9). A **`target`** can be set to a number of preset values: **`"_blank"`** loads the page into a new browser window, **`"_self"`** loads the page into the frame in which the anchor element appears and **`"_top"`** loads the page into the full browser window (i.e., removes the **`frameset`**).

5.10 Nested `framesets`

You can use the **`frameset`** element to create more complex layouts in a Web page by nesting **`frameset`**s, as in Fig. 5.11. The nested **`frameset`** in this example displays the XHTML documents in Fig. 5.7, Fig. 5.8 and Fig. 5.10.

The outer frameset element (lines 23–41) defines two columns. The left frame extends over the first 110 pixels from the left edge of the browser and the right frame occupies the rest of the window's width. The **frame** element on line 24 specifies that the document **nav.html** (Fig. 5.10) will be displayed in the left column.

Lines 28–31 define a nested **frameset** element for the second column of the outer frameset. This **frameset** defines two rows. The first row extends 175 pixels from the top of the browser window, as indicated by **rows = "175,*"**. The second row occupies the remainder of the browser window's height. The **frame** element at line 29 specifies that the first row of the nested **frameset** will display **picture.html** (Fig. 5.7). The **frame** element at line 30 specifies that the second row of the nested **frameset** will display **main.html** (Fig. 5.9).

```
1  <?xml version = "1.0"?>
2  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
3     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
4
5  <!-- Fig. 5.11: index2.html -->
6  <!-- XHTML Frames II        -->
7
8  <html xmlns = "http://www.w3.org/1999/xhtml">
9     <head>
10       <title>Internet and WWW How to Program - Main</title>
11
12       <meta name = "keywords" content = "Webpage, design,
13          XHTML, tutorial, personal, help, index, form,
14          contact, feedback, list, links, frame, deitel" />
15
16       <meta name = "description" content = "This Web site will
17          help you learn the basics of XHTML and Web page design
18          through the use of interactive examples
19          and instruction." />
20
21    </head>
22
23    <frameset cols = "110,*">
24       <frame name = "leftframe" src = "nav.html" />
25
26       <!-- nested framesets are used to change the -->
27       <!-- formatting and layout of the frameset  -->
28       <frameset rows = "175,*">
29          <frame name = "picture" src = "picture.html" />
30          <frame name = "main" src = "main.html" />
31       </frameset>
32
33       <noframes>
34          <p>This page uses frames, but your browser does not
35          support them.</p>
36
37          <p>Please, <a href = "nav.html">follow this link to
38          browse our site without frames</a>.</p>
39       </noframes>
```

Fig. 5.11 Framed Web site with a nested frameset (part 1 of 2).

```
40
41      </frameset>
42  </html>
```

Fig. 5.11 Framed Web site with a nested frameset (part 2 of 2).

Testing and Debugging Tip 5.2

When using nested **`frameset`** *elements, indent every level of* **`<frame>`** *tag. This practice makes the page clearer and easier to debug.*

In this chapter, we presented XHTML for marking up information in tables, creating forms for gathering user input, linking to sections within the same document, using **`<meta>`** tags and creating frames. In Chapter 6, we build upon the XHTML introduced in this chapter by discussing how to make Web pages more visually appealing with Cascading Style Sheets.

5.11 Internet and World Wide Web Resources

`courses.e-survey.net.au/xhtml/index.html`
The *Web Page Design - XHTML* site provides descriptions and examples for various XHTML features, such as links, tables, frames, forms, etc. Users can e-mail questions or comments to the Web Page Design support staff.

`www.vbxml.com/xhtml/articles/xhtml_tables`
The *VBXML.com* Web site contains a tutorial on creating XHTML tables.

`www.webreference.com/xml/reference/xhtml.html`
This Web page contains a list of the frequently used XHTML tags, such as header tags, table tags, frame tags and form tags. It also provides a description of each tag.

SUMMARY

- 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 check box. 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.
- Image maps designate certain sections of an image as links. These links are more properly called hotspots.
- Image maps are defined with **map** elements. Attribute **id** identifies the image map. Hotspots are defined with the **area** element. Attribute **href** specifies the link's target. Attributes **shape** and **coords** specify the hotspot's shape and coordinates, respectively, and **alt** provides alternate text.
- One way that search engines catalog pages is by reading the **meta** elements' contents. Two important attributes of the **meta** element are **name**, which identifies the type of **meta** element and ***content***, which provides information a search engine uses to catalog a page.
- Frames allow the browser to display more than one XHTML document simultaneously. The **frameset** element informs the browser that the page contains frames. Not all browsers support frames. XHTML provides the **noframes** element to specify alternate content for browsers that do not support frames.
- You can use the **frameset** element to create more complex layouts in a Web page by nesting **frameset**s.

TERMINOLOGY

action attribute
area element
border attribute
browser request
<caption> tag
checkbox
checked attribute
col element
colgroup element
cols attribute
colspan attribute
coords element
form
form element
frame element
frameset element
header cell
hidden input element
hotspot
href attribute
image map
img element
input element
internal hyperlink
internal linking
map element
maxlength attribute
meta element
method attribute
name attribute
navigational frame
nested **frameset** element
nested tag
noframes element
password box
"radio" (attribute value)
rows attribute (**textarea**)
rowspan attribute (**tr**)
selected attribute
size attribute (**input**)
table element
target = "_blank"
target = "_self"
target = "_top"
tbody element
td element
textarea
textarea element
tfoot (table foot) element
<thead>...</thead>
tr (table row) element
type attribute
usemap attribute
valign attribute (**th**)
value attribute
Web server
XHTML form
x-*y*-coordinate

SELF-REVIEW EXERCISES

5.1 State whether the following statements are *true* or *false*. If *false*, explain why.

a) The width of all data cells in a table must be the same.
b) Framesets can be nested.
c) You are limited to a maximum of 100 internal links per page.
d) All browsers can render **frameset**s.

5.2 Fill in the blanks in each of the following statements:

a) Assigning attribute **type** _____ in an **input** element inserts a button that, when clicked, clears the contents of the form.
b) The layout of a **frameset** is set by including the _____ attribute or the _____ attribute inside the **<frameset>** tag.
c) The _____ element marks up a table row.
d) _____ are used as masking characters in a password box.
e) The common shapes used in image maps are _____, _____ and _____.

5.3 Write XHTML markup to accomplish each of the following:

a) Insert a framed Web page, with the first frame extending 300 pixels across the page from the left side.
b) Insert a table with a border of 8.
c) Indicate alternate content to a **frameset**.
d) Insert an image map in a page using **deitel.gif** as an image and **map** with **name = "hello"** as the image map, and set the **alt** text to "**hello**".

ANSWERS TO SELF-REVIEW EXERCISES

5.1 a) False. You can specify the width of any column, either in pixels or as a percentage of the table width. b) True. c) False. You can have an unlimited number of internal links. d) False. Some browsers are unable to render a **frameset** and must therefore rely on the information that you include inside the **<noframes>**…**</noframes>** tags.

5.2 a) **"reset"**. b) **cols**, **rows**. c) **tr**. d) asterisks. e) **poly** (polygons), **circle**s, **rect** (rectangles).

5.3 a)
```
<frameset cols = "300,*">...</frameset>
```
b)
```
<table border = "8">...</table>
```
c)
```
<noframes>...</noframes>
```
d)
```
<img src = "deitel.gif" alt = "hello" usemap = "#hello" />
```

EXERCISES

5.4 Categorize each of the following as an element or an attribute:

a) **width**
b) **td**
c) **th**
d) **frame**
e) **name**
f) **select**
g) **type**

5.5 What will the **frameset** produced by the following code look like? Assume that the pages referenced are blank with white backgrounds and that the dimensions of the screen are 800 by 600. Sketch the layout, approximating the dimensions.

```
<frameset rows = "20%,*">
   <frame src = "hello.html" name = "hello" />
      <frameset cols = "150,*">
         <frame src = "nav.html" name = "nav" />
         <frame src = "deitel.html" name = "deitel" />
      </frameset>
</frameset>
```

5.6 Write the XHTML markup to create a frame with a table of contents on the left side of the window, and have each entry in the table of contents use internal linking to scroll down the document frame to the appropriate subsection.

5.7 Create XHTML markup that produces the table shown in Fig. 5.12. Use **** and **** tags as necessary. The image (**camel.gif**) is included in the Chapter 5 examples directory on the CD-ROM that accompanies this book.

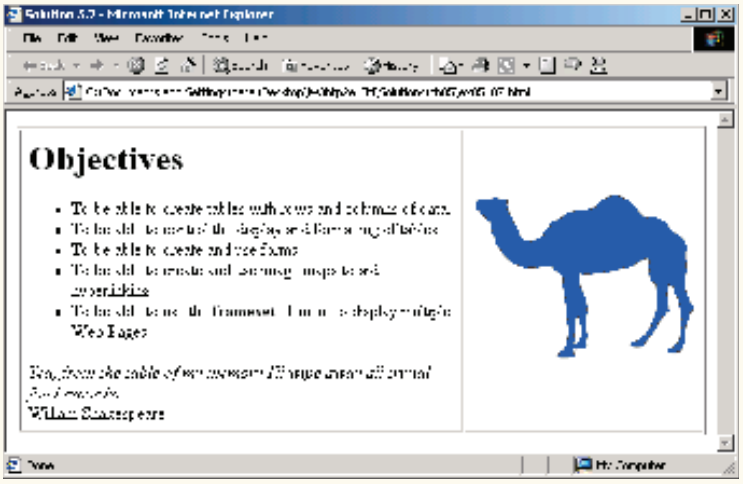

Fig. 5.12 XHTML table for Exercise 5.7.

5.8 Write an XHTML document that produces the table shown in Fig. 5.13.

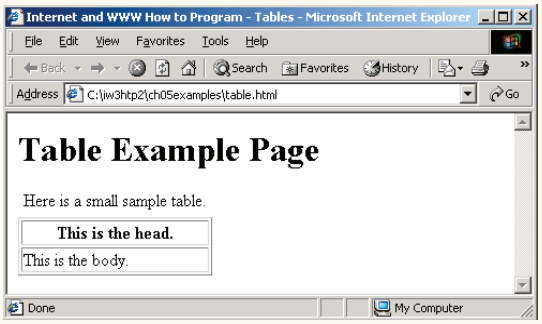

Fig. 5.13 XHTML table for Exercise 5.8.

5.9 A local university has asked you to create an XHTML document that allows potential students to provide feedback about their campus visit. Your XHTML document should contain a form with text boxes for a name, address and e-mail. Provide check boxes that allow prospective students to indicate what they liked most about the campus. These check boxes should include: students, location, campus, atmosphere, dorm rooms and sports. Also, provide radio buttons that ask the prospective student how they became interested in the university. Options should include: friends, television, Internet and other. In addition, provide a text area for additional comments, a submit button and a reset button.

5.10 Create an XHTML document titled "How to Get Good Grades." Use **`<meta>`** tags to include a series of keywords that describe your document.

5.11 Create an XHTML document that displays a tic-tac-toe table with player X winning. Use **`<h2>`** to mark up both Xs and Os. Center the letters in each cell horizontally. Title the game using an **`<h1>`** tag. This title should span all three columns. Set the table border to one.