Creating a Selection List

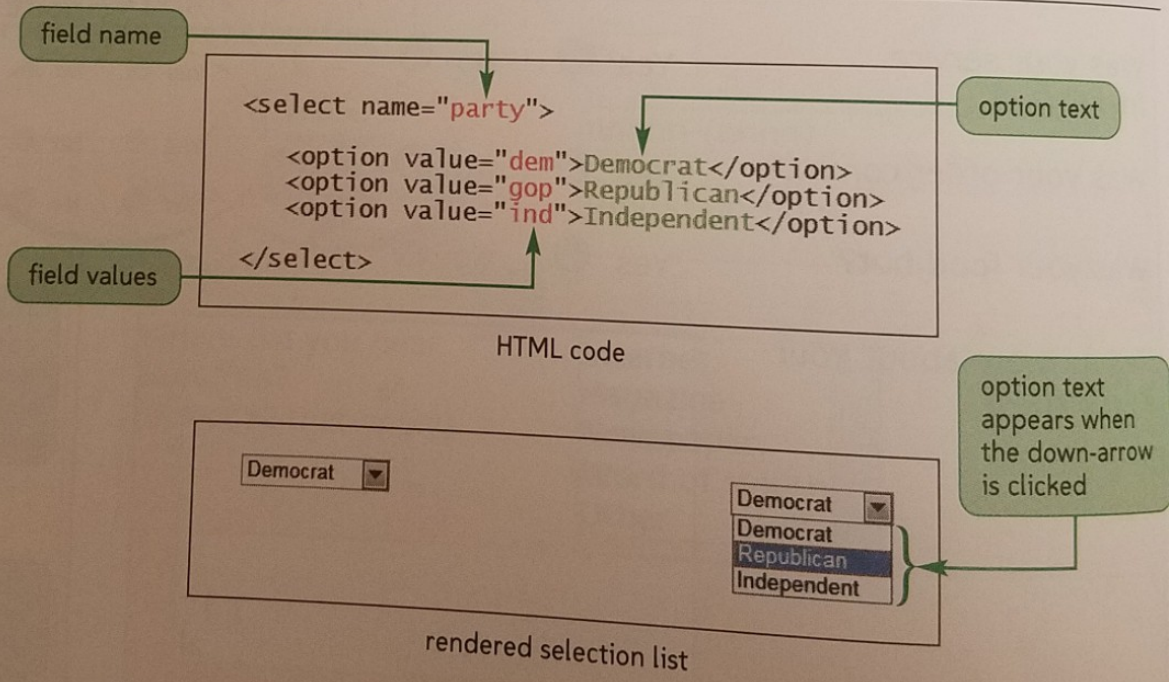
The next part of the survey form records how customers place their orders from Red Ball Pizza. A customer order can be placed in one of four ways: pickup; delivery; dine in; or in the case of pizzas, uncooked pizzas that customers can take home and bake. Alice doesn't want customers to enter their order types into an input box because different customers will enter this information in different ways, and the large variety of spellings and text will make it difficult to group and analyze the survey results. Instead, she wants each user to select the order type from a predetermined group of options. This can be accomplished using a selection list.

A selection list is a list box that presents users with a group of possible field values for the data field. It's created using the HTML code

```
<select name="name" id="id">
   <option value="value1">text1</option>
   <option value="value2">text2</option>
   ...
</select>
```

where the `name` and `id` attributes provide the name of the data field and identify the selection list control, respectively; *value1*, *value2*, etc. are the possible values of the data field; and *text1*, *text2*, etc. are the text of the entries in the selection list that users see on the Web form. Figure 6-25 shows an example of a selection list used to record each user's political affiliation.

Creating a selection list



Notice that the field value does not have to match the option text. In most cases, the option text will be more expansive and descriptive to make it easier for users, while the corresponding value will be brief and succinct for use with the server program analyzing the form data.

REFERENCE

Creating a Selection List

- To create a selection list, add the elements

```
<select name="name">
   <option value="value1">text1</option>
   <option value="value2">text2</option>
   ...
</select>
```

to the Web form, where *name* is the name of the field associated with the selection list; *value1*, *value2*, etc. are the possible field values; and *text1*, *text2*, etc. are the entries displayed in the selection list.

- To specify the default value, add the following attribute to one of the *option* elements:

```
selected="selected"
```

- To set the number of options displayed at one time in the selection list, add the attribute

```
size="value"
```

to the `select` element, where *value* is the number of options displayed in the selection list at any one time.

- To allow users to make multiple selections, add the attribute

```
multiple="multiple"
```

to the `select` element.

You'll add a selection list to the Red Ball Pizza survey form to record the type of order placed by the customer, storing the value in the `ordertype` field. The program that will analyze these results will use the field values `type1`, `type2`, `type3`, or `type4`, but the option text in the selection list will read *Carry out*, *Delivery*, *Dine in*, and *Take and bake*, respectively.

To create the selection list:

1. Return to the **survey.htm** file in your text editor and scroll down to the bottom of the second field set.
2. Directly before the closing `</fieldset>` tag, add the following code (see Figure 6-26):

```
<label for="ordertype">Order type</label>
<select name="ordertype" id="ordertype">
   <option value="type1">Carry out</option>
   <option value="type2">Delivery</option>
   <option value="type3">Dine in</option>
   <option value="type4">Take and bake</option>
</select>
```

Creating the ordertype selection list

label text

```
      <label for="ordertype">Order type</label>
      <select name="ordertype" id="ordertype">
         <option value="type1">Carry out</option>
         <option value="type2">Delivery</option>
         <option value="type3">Dine in</option>
         <option value="type4">Take and bake</option>
      </select>

   </fieldset>
</form>
```

selection list options

3. Save your changes to the file.

You'll also want to set the style of the `select` element so that, like the input boxes you created in the last session, it's floated alongside its label, and its font size and margin space are set to match the layout of the survey form.

4. Go to the **forms.css** file in your text editor. At the bottom of the file, add the following code as shown in Figure 6-27:

```
/* Selection list styles */

select {
   display: block;
   float: left;
   font-size: 0.9em;
   margin: 7px 0px;
}
```

Style rule for the select element

splays the selection list as a block

loats the selection list on the left

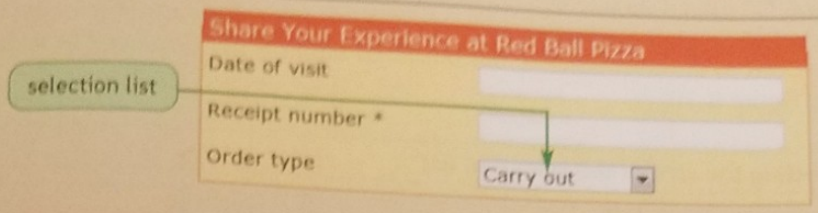
sets the font size to 0.9 em

n space around the selection list

```
input#state {
   width: 50px;
}


/* Selection list styles */

select {
   display: block;
   float: left;
   font-size: 0.9em;
   margin: 7px 0px;
}
```

5. Save your changes to the style sheet file and then open the **survey.htm** file in your Web browser. The survey form now displays a selection list for the type of order (see Figure 6-28). Click the selection list arrow and verify that all of the order type options are shown.

6-28 **Order type selection list**



Trouble? Depending on your browser, the selection list may appear slightly different from the one shown in Figure 6-28.

The first option in a selection list is the field's default value. To specify a different default value and to display a different option from the selection list, add the `selected` attribute to the `option` element as follows:

```
<option selected="selected" value="value">text</option>
```

Browsers also will accept the `selected` attribute without an attribute value, appearing as follows:

```
<option selected value="value">text</option>
```

However, this syntax is not supported in XHTML documents. To be consistent across the different markup languages, this book always includes an attribute value, even when that attribute value does nothing more than repeat the attribute name.

Alice knows that most of the survey respondents dine in at the restaurant. Although the options in the Order type selection list are displayed in alphabetical order, she would like the *Dine in* option selected by default.

To specify the default value for the selection list:

1. Return to the **survey.htm** file in your text editor and add the `selected="selected"` attribute to the *Dine in* option (see Figure 6-29).

Specifying the selected option

```
<label for="ordertype">Order type</label>
<select name="ordertype" id="ordertype">
   <option value="type1">Carry out</option>
   <option value="type2">Delivery</option>
   <option value="type3" selected="selected">Dine in</option>
   <option value="type4">Take and bake</option>
</select>
```

2. Save your changes to the file and then reopen **survey.htm** in your Web browser. Verify that the *Dine in* option is preselected in the order type list.

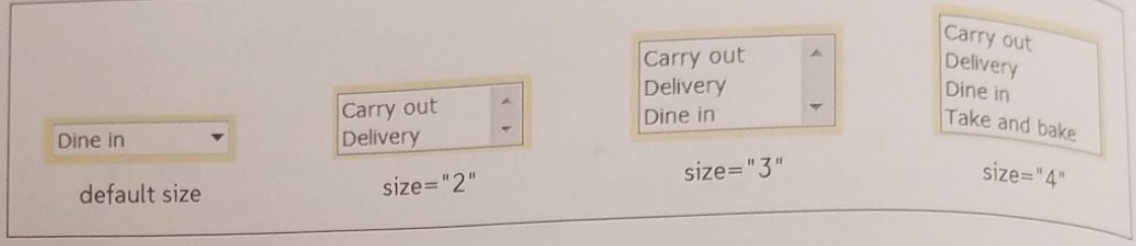
Setting the Size of the Selection List

By default, selection lists display only the currently selected option value. You can change the number of options displayed by applying the `size` attribute

```
<select size="value"> ... </select>
```

to the `select` element, where *value* is the number of options that the selection list displays at one time. By specifying a `size` value greater than 1, you change the selection list from a drop-down list box to a list box with a scroll bar that allows users to scroll through the selection options. If you set the `size` attribute equal to the number of options in the selection list, the scroll bar either is not displayed or is dimmed, as shown in Figure 6-30.

Setting the size of the selection list



Alice has another selection list to add to the survey form, recording where the customer heard of Red Ball Pizza. The survey presents the user with five options: the Internet, a magazine, a newspaper, word of mouth, or other. Alice wants you to display all of the options by setting the value of the `size` attribute to 5.

To set the selection list size:

1. Return to the **survey.htm** file in your text editor and add the following label and selection list directly below the `email` field (see Figure 6-31):

```
<label>Where did you hear about us?</label>
<select name="infoSrc" id="infoSrc" size="5">
   <option value="internet">Internet</option>
   <option value="mag">Magazine</option>
   <option value="news">Newspaper</option>
   <option value="word">Word of Mouth</option>
   <option value="other">Other</option>
</select>
```

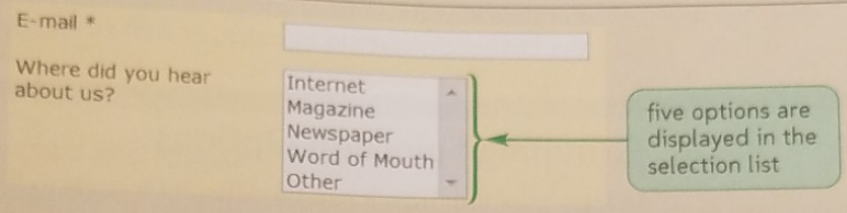
Creating the infoSrc selection list

setting the size of the selection list to five options

```
<label for="email">E-mail *</label>
<input name="email" id="email" />

<label>Where did you hear about us?</label>
<select name="infoSrc" id="infoSrc" size="5">
   <option value="internet">Internet</option>
   <option value="mag">Magazine</option>
   <option value="news">Newspaper</option>
   <option value="word">Word of Mouth</option>
   <option value="other">Other</option>
</select>
```

2. Save your changes to the file and then reopen **survey.htm** in your Web browser. As shown in Figure 6-32, the selection list appears with all five options displayed.

6-32 List box for listing the information source



Trouble? Depending on your browser, you might not see a scroll bar next to the list of options.

Allowing for Multiple Selections

In the code you just entered, customers were limited to a single option. However, Alice is aware that a customer could have heard about Red Ball Pizza from more than one source. She would like customers to be able to select more than one option if applicable. Multiple selections can be applied to a selection list by adding the `multiple` attribute to the `select` element as follows:

```
<select multiple="multiple"> … </select>
```

As with the `selected` attribute discussed earlier, you also can apply the `multiple` attribute without an attribute value and most browsers will interpret it correctly.

There are two ways for users to select multiple items from a selection list. For noncontiguous selections, users can press and hold the Ctrl key (or the command key on a Mac) while making the selections. For a contiguous selection, users can select the first item, press and hold the Shift key, and then select the last item in the range. This selects the two items as well as all the items between them.

If you decide to use a multiple selection list in a form, be aware that the form sends a name/value pair to the server for each option the user selects from the list. Verify that your server-based program can handle a single field with multiple values before using a multiple selection list.

You'll add the ability to select multiple options to the `infoSrc` field you just created.

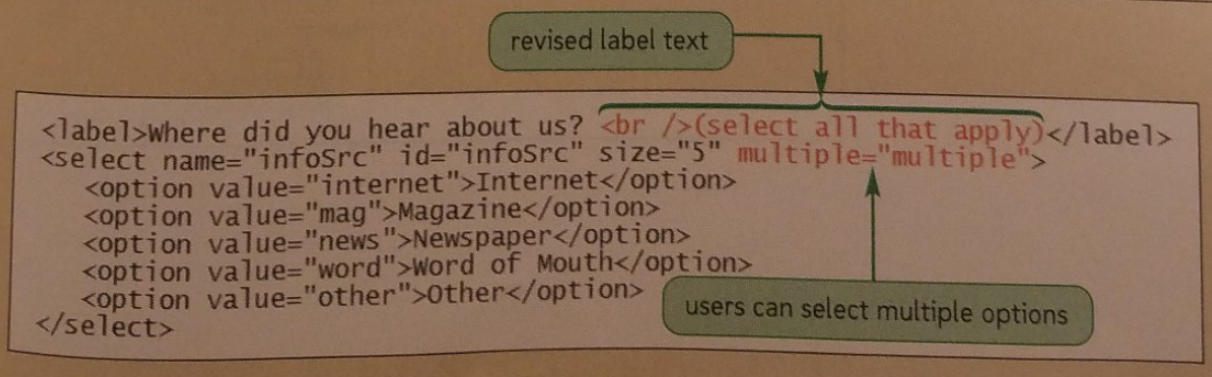
To allow for multiple selections:

1. Return to the **survey.htm** file in your text editor and then add the following text to the `label` element for the `infoSrc` selection list:

```
<br />(select all that apply)
```

2. Add the attribute `multiple="multiple"` to the `select` element. Figure 6-33 highlights the newly added code.

Allowing for multiple selections



3. Save your changes to the file and then reopen **survey.htm** in your Web browser. Verify that you can now select multiple items from the information source list using the ctrl+click, command+click, or shift+click keyboard and mouse combinations.

Grouping Selection Options

In long selection lists, it can be difficult for users to locate a particular option value. You can organize selection list options by placing them in option groups using the `optgroup` element

```
<select>
   <optgroup label="label1">
      <option>text1</option>
      <option>text2</option>
...
   </optgroup>
   <optgroup label="label2">
      <option>text1</option>
      <option>text2</option>
...
   </optgroup>
</select>
```

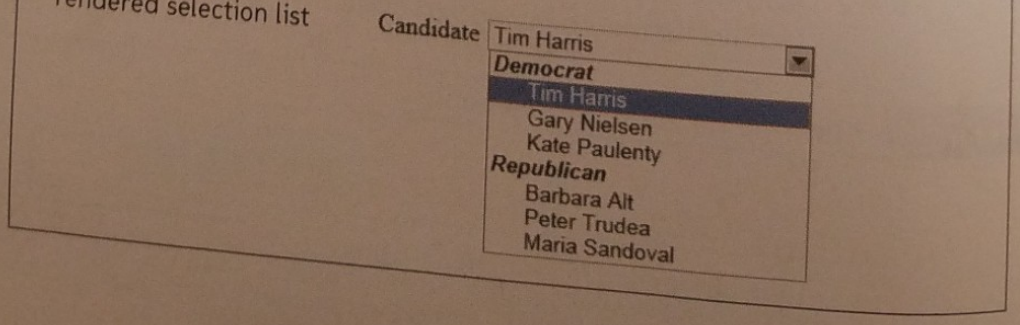
where *label1*, *label2*, and so forth are the labels for the different groups of options. The text of the label appears in the selection list above each group of items but is not a selectable item from the list. Figure 6-34 shows an example of a selection list in which the options are divided into two groups.

Organizing a selection list with option groups

HTML code

```
<label for="party">Candidate</label>
<select name="party">

   <optgroup label="Democrat">
      <option value="d1">Tim Harris</option>
      <option value="d2">Gary Nielsen</option>
      <option value="d3">Kate Paulenty</option>
   </optgroup>

   <optgroup label="Republican">
      <option value="r1">Barbara Alt</option>
      <option value="r2">Peter Trudea</option>
      <option value="r3">Maria Sandoval</option>
   </optgroup>

</select>
```

rendered selection list



The appearance of the option group label is determined by the browser. You can apply a style to an entire option group including its label, but there is no CSS style to change the appearance of the option group label alone.

Alice does not need you to use an option group for her survey form because the number of options is so small.

INSIGHT

Creating Passwords and Hidden Fields

Fields and field values are not always visible to users. For sensitive data such as passwords or credit card numbers, you can display an input box using the `password` data type

```
<input name="name" type="password" />
```

where *name* is the field name. Any information that a user enters will be displayed as a series of dots or asterisks, protecting the information from prying eyes.

Another way of hiding information is with a hidden field, which is created using an `input` element with the hidden data type, as follows

```
<input name="name" type="hidden" value="value" />
```

where *value* is the value stored in the field. With a hidden field, both the field value and the input control are hidden from the user. Hidden fields are used for fields that have a predefined value that will be used by the script processing the Web form. Even though hidden fields are not displayed by browsers, the field values still can be read by examining the source code; for this reason, you should not put any sensitive information in a hidden field.

Creating Option Buttons

In the next part of the form, Alice wants to ask customers general questions about their experiences at the restaurant. She wants to know whether the service was friendly, whether orders were recorded correctly, and if the food was delivered hot. She suggests that you present these questions using option buttons.

Option buttons, also called radio buttons, are like selection lists in that they limit users to a set of possible values; but unlike selection lists, the options appear as separate control elements on the form. Option buttons are created using the `input` element with the `type` attribute set to a value of `radio` as follows

```
<input type="radio" name="name" value="value1" />
<input type="radio" name="name" value="value2" />
<input type="radio" name="name" value="value3" />
...
```

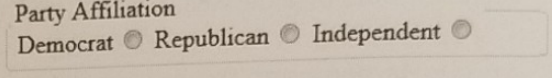
where *name* is the name of the data field associated with the option buttons, and *value1*, *value2*, *value3*, etc. are the field values associated with each option. When multiple option buttons are applied to the same data field, browsers treat them as a group, and selecting one option button automatically deselects all of the others. Figure 6-35 shows an example of a Web form that uses an option button group to indicate political party affiliations.

Creating a group of option buttons

HTML code

```
<fieldset>
    <legend>Party Affiliation<legend>

    <label for="demOption">Democrat</label>
    <input type="radio" name="party" id="demOption" value="dem" />

    <label for="repOption">Republican</label>
    <input type="radio" name="party" id="repOption" value="rep" />

    <label for="indOption">Independent</label>
    <input type="radio" name="party" id="indOption" value="ind" />

</fieldset>
```

rendered option buttons



In this sample code, all of the option buttons are associated with the `party` field with values of `dem`, `rep`, and `ind`, respectively. The option buttons are organized within a field set, and each label within the field set is linked to a specific option button control using the `for` and `id` attributes. When you link a label to an option button, users can select the option by clicking either the label or the option button. For example, clicking on the label text *Democrat* would cause a browser to select the `demoOption` option button.

REFERENCE

Creating a Group of Option Buttons

- To create a group of option buttons associated with a single field, add the `input` elements

```
<input type="radio" name="name" value="value1" />
<input type="radio" name="name" value="value2" />
<input type="radio" name="name" value="value3" />
...
```

where *name* is the name of the data field, and *value1*, *value2*, *value3*, etc. are the field values associated with each option.

- To specify the default option, add the `checked` attribute to the `input` element as follows:

```
checked="checked"
```

You'll create option buttons now for the `serviceFriendly` field to record whether a customer was treated well by the Red Ball Pizza staff. To keep the control elements for the option buttons organized as a group, you'll nest them and their labels within a `fieldset` element with the class name `optionGroup`.

To create option buttons:

1. Return to the **survey.htm** file in your text editor and then scroll down to the second field set. Directly after the `ordertype` selection list, add the following code (see Figure 6-36):

```
<label>Was your service friendly?</label>
<fieldset class="optionGroup">
   <label for="sYes">Yes</label>
   <input type="radio" name="serviceFriendly" id="sYes"
          value="yes" />

   <label for="sNo">No</label>
   <input type="radio" name="serviceFriendly" id="sNo"
          value="no" />
</fieldset>
```

Figure 6-36 Option button group for the serviceFriendly field

set containing the el and option buttons

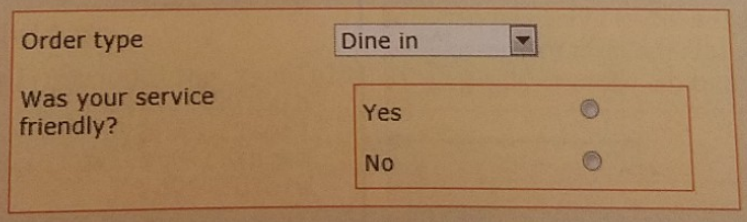
l associated with the on button control

on button for the ceFriendly field

```
   <option value="type4">Take and bake</option>
</select>

<label>Was your service friendly?</label>
<fieldset class="optionGroup">
   <label for="sYes ">Yes</label>
   <input type="radio" name="serviceFriendly" id="sYes"
          value="yes" />

   <label for="sNo">No</label>
   <input type="radio" name="serviceFriendly" id="sNo"
          value="no" />
</fieldset>
```

2. Save your changes to the file and then reopen **survey.htm** in your Web browser. Figure 6-37 shows the current appearance of the option buttons.

Figure 6-37 Rendered option buttons



3. Click each option button and confirm that clicking one option button deselects the other. Also verify that when you click the labels next to the option buttons, option buttons become selected.

The appearance of the option buttons and the field set box is partly based on the C styles you created for the `fieldset` and `input` elements in the last session. The optic buttons take up more space than necessary and you think they would look better if th were all on a single line. To accomplish this, you'll create style rules to display the la and option buttons as inline elements, reduce their widths, resize their margins, and vent them from floating. You'll also remove the border from the `fieldset` element th contains the option button labels and controls.

To revise the styles for the option button group:

1. Go to the **forms.css** file in your text editor.
2. At the bottom of the file, insert the following style rules as shown in Figure 6-38:

```
/* Option button styles */

fieldset.optionGroup {
   border-width: 0px;
}

fieldset.optionGroup label {
   display: inline;
   float: none;
   margin: 0px 3px 0px 0px;
   width: 30px;
}

fieldset.optionGroup input {
   display: inline;
   float: none;
   margin: 0px 20px 0px 0px;
   width: 20px;
}
```

38 Style rules for the option button group

removes the border from the field set

displays the labels inline without floating; sets the right margin to 3 pixels and the width to 30 pixels

displays the input controls inline without floating; sets the right margin and width to 20 pixels

```
/* Option button styles */

fieldset.optionGroup {
   border-width: 0px;
}

fieldset.optionGroup label {
   display: inline;
   float: none;
   margin: 0px 3px 0px 0px;
   width: 30px;
}

fieldset.optionGroup input {
   display: inline;
   float: none;
   margin: 0px 20px 0px 0px;
   width: 20px;
}
```

3. Save your changes to the file and then reopen the **survey.htm** file in your We browser. Figure 6-39 shows the revised appearance of the option button grou

39 Revised appearance of the option button group

Order type: Dine in

Was your service friendly? Yes No

Trouble? In some browsers, the label text will not wrap to a new line.

There are two other option button groups that Alice wants in the survey form: one to find out whether the customer's order was delivered correctly, and the other to find out if the food was presented hot. You'll add these option button groups now.

To add the remaining option button groups:

1. Return to the **survey.htm** file in your text editor.
2. Directly below the `fieldset` element for the `serviceFriendly` field, add the following HTML code (see Figure 6-40):

```
<label>Was your order correct?</label>
<fieldset class="optionGroup">
   <label for="oYes">Yes</label>
   <input type="radio" name="orderCorrect" id="oYes"
          value="yes" />

   <label for="oNo">No</label>
   <input type="radio" name="orderCorrect" id="oNo"
          value="no" />
</fieldset>

<label>Was your food hot?</label>
<fieldset class="optionGroup">
   <label for="hotYes">Yes</label>
   <input type="radio" name="foodHot" id="hotYes"
          value="yes" />

   <label for="hotNo">No</label>
   <input type="radio" name="foodHot" id="hotNo"
          value="no" />
</fieldset>
```

6-40 Option button groups for the orderCorrect and foodHot fields

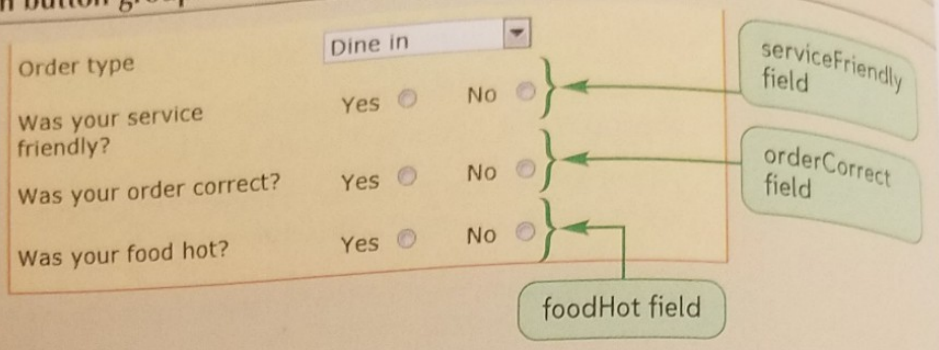
field set for the orderCorrect field

field set for the foodHot field

```
   <input type="radio" name="serviceFriendly" id="sNo"
          value="no" />
</fieldset>

<label>Was your order correct?</label>
<fieldset class="optionGroup">
   <label for="oYes">Yes</label>
   <input type="radio" name="orderCorrect" id="oYes"
          value="yes" />

   <label for="oNo">No</label>
   <input type="radio" name="orderCorrect" id="oNo"
          value="no" />
</fieldset>

<label>Was your food hot?</label>
<fieldset class="optionGroup">
   <label for="hotYes">Yes</label>
   <input type="radio" name="foodHot" id="hotYes"
          value="yes" />

   <label for="hotNo">No</label>
   <input type="radio" name="foodHot" id="hotNo"
          value="no" />
</fieldset>
```

3. Save your changes to the file and then reopen the **survey.htm** file in your Web browser. Figure 6-41 shows all of the option button groups in the survey form.

Completed option button groups



By default, an option button is unselected; however, you can set an option button to be selected when a form opens by adding the `checked` attribute to the `input` element as follows:

```
<input type="radio" name="name" checked="checked" />
```

You also can enter the `checked` attribute without an attribute value, and most browsers will be able to interpret your code.

Creating a Text Area Box

Alice wants the survey form to include a place where customers can enter extended comments about Red Ball Pizza. She wants customers to be able to enter several lines of text. Because an input box is limited to a single line of text, it would not be appropriate to enter those comments in an input box. Instead, you can create a control element that allows for extended text entries using the `textarea` element

```
<textarea name="name">
   text
</textarea>
```

where *`text`* is default text that is placed in the text area box. You do not have to specify default text and can leave the text box empty. Many browsers also support the `placeholder` attribute introduced in the last session to provide additional hints to users about what to enter into the text box.

Browsers determine the default size of a text area box. Most browsers create a box that is about 20 characters long and two or three lines high. You can set the dimensions of a text area box using the CSS `width` and `height` style properties. HTML also supports the `row` and `cols` attributes to set the size of the box, as follows

```
<textarea rows="value" cols="value"> ... </textarea>
```

where the `rows` attribute specifies the number of lines in the text area box and the `cols` attribute specifies the number of characters per line.

As you type text into a text area box, the text automatically wraps to a new line as it extends beyond the box's width. If more text is entered into a box than can be displayed, the browser automatically adds horizontal and vertical scroll bars to the box. You can determine whether the locations of line wrapping are included in the field value by using the `wrap` attribute

```
<textarea wrap="type"> ... </textarea>
```

where *type* is either `hard` or `soft`. In a hard wrap, information about where the text begins a new line is included with the data field value, while in a soft wrap this information is not included. When a hard wrap is used, the `cols` attribute also needs to be specified. Many browsers also support the `off` wrap type to prevent browsers from wrapping text within a text area box, though it is not part of any HTML specification. The default value of the `wrap` attribute is `soft`.

REFERENCE

Creating a Text Area Box

- To create a text area box for multiple lines of text, use the element

```
<textarea name="name">
   text
</textarea>
```

where *name* is the name of the field associated with the text area box and *text* is the default text that appears in the box.

- To specify the dimensions of the box, add the attributes

```
rows="value" cols="value"
```

to the `textarea` element, where the `rows` attribute specifies the number of lines in the text area box and the `cols` attribute specifies the number of characters per line.

- To specify how the field value should handle wrapped text, use the attribute

```
wrap="type"
```

where *type* is either `hard` (to include the locations of the line wraps) or `soft` (to ignore line wrap locations).

You'll add a `textarea` element to the survey form to store the customer comments. You'll also set the width and the height of the element using a CSS style rule.

To create the comments text area box:

1. Return to the **survey.htm** file in your text editor.
2. Directly below the `fieldset` element for the `foodHot` option group you just created, enter the following code (see Figure 6-42):

```
<label for="comments">Tell us more about your experience!</label>
<textarea name="comments" id="comments"></textarea>
```

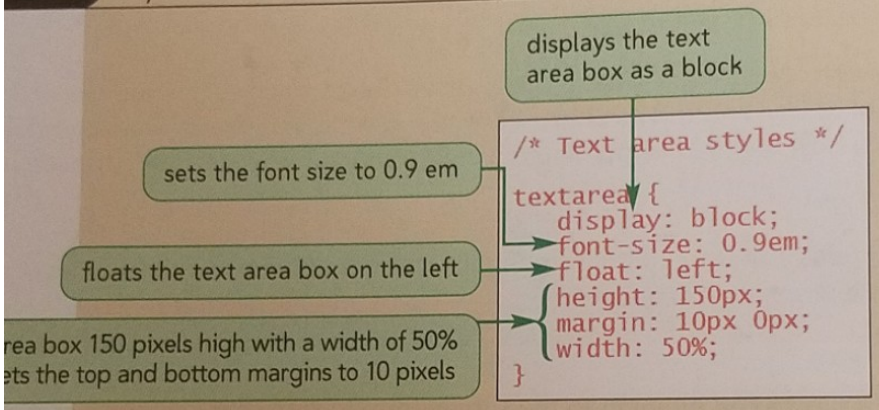
Creating a text area control

```
    <label for for="hotNo">No</label>
    <input type="radio" name="foodHot" id="hotNo"
           value="no" />
</fieldset>

<label for= "comments ">Tell us more about your experience!</label>
<textarea name="comments" id="comments"></textarea>

</fieldset>
```

3. Save your changes to the file and then return to the **forms.css** file in your text editor.
4. At the bottom of the style sheet file, insert the following style rule as shown in Figure 6-43:

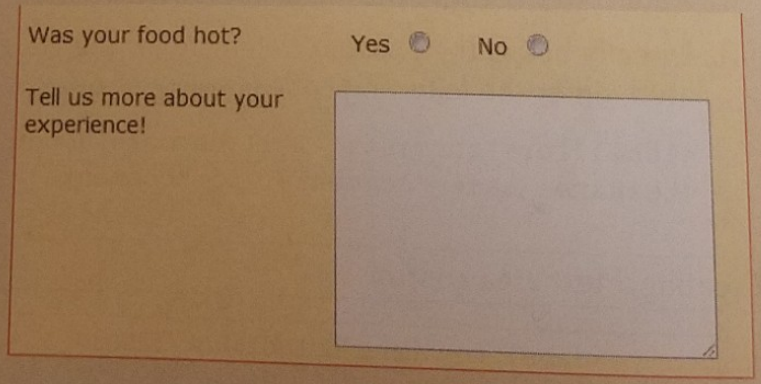
```
/* Text area styles */

textarea {
   display: block;
   font-size: 0.9em;
   float: left;
   height: 150px;
   margin: 10px 0px;
   width: 50%;
}
```

43 **Style rule for the text area box**



5. Save your changes to the file and then reopen the **survey.htm** file in your Web browser. Figure 6-44 shows the newly added text area box.

4 **Text area box**



6. Type some sample text into the text area box and verify that the text wraps to a new line as you exceed the width of the box.

Trouble? Line wraps do not occur in the middle of words. If you find your samp text is not wrapping to a new line, make sure you are entering individual words rather than a long character string.

INSIGHT

Tab Indexing and Autofocus

Typically, users navigate through a Web form using the Tab key, which moves the cursor from one field to another in the order that the field tags are entered into the HTML file.

You can specify an alternate order by adding the `tabindex` attribute to any control element in your form. When each element is assigned a tab index number, the cursor moves through the fields from the lowest index number to the highest. For example, to assign the tab index number 1 to the `name` field from the survey form, you would enter the following `tabindex` attribute to the control element:

```
<input name="name" tabindex="1" />
```

This code would ensure that the cursor is in the `name` field when the form is first opened. (Fields with 0 or negative tab indexes are omitted from the tab order entirely.)

Another way to ensure that a particular field is selected when a Web form is initially opened is to use the `autofocus` attribute. The HTML code

```
<input name="name" autofocus="autofocus" />
```

automatically places the cursor in the input box for the `name` field when the form is loaded by a browser. The `autofocus` attribute was introduced in HTML5 and thus might not be supported by older browsers.

Older browsers that do not support tab indexing or the `autofocus` attribute simply ignore them and open a file without giving the focus to any control element. When a user tabs through the form, the tab order will reflect the order of the items in the HTML file.

Creating Check Boxes

A survey form like the one you're creating for Red Ball Pizza serves two purposes. One, of course, is to receive customer feedback; the other is to remain in contact with the consumer base. Red Ball Pizza has an e-mail newsletter that it sends out to subscribers, detailing the latest news about the restaurant and informing patrons about upcoming events and special deals. Alice wants to give customers filling out the survey form a way of subscribing to the newsletter. This can be done using a check box.

You use a check box control in situations where you are verifying the presence or absence of something; in this case, whether or not a customer is interested in receiving e-mail from the restaurant. Check boxes are created using the `input` element with the `type` attribute set to `checkbox`, as follows:

```
<input type="checkbox" name="name" value="value" />
```

The `value` attribute contains the value of the field when the check box is checked. If no value is provided, the value `on` is used by default. For example, the following code creates a check box for determining whether a user is a member of the Democratic party

```
<label for="dem">Democrat?</label>
<input type="checkbox" name="dem" id="dem" value="yes" />
```

If the check box is selected, the browser will submit a name-value pair of `dem-yes` to the script running on the Web server. A name-value pair is sent to the server only when the check box is checked by the user. If it is not checked, then nothing is sent to the serve

By default, check boxes are not selected. To preselect a check box, add the `checked` attribute to the `input` element as follows:

```
checked="checked"
```

As with the `multiple` and `selected` attributes, you also can use the `checked` attribute without an attribute value.

REFERENCE

Creating a Check Box

- To create a check box, add the element

```
<input type="checkbox" name="name" value="value" />
```

to the Web form, where *name* is the name of the data field and *value* is the data field value if the check box is selected.
- To specify that a check box is selected by default, add the following attribute to the `input` element:

```
checked="checked"
```

You'll add a check box below the two field sets asking customers whether they would like to receive the Red Ball Pizza e-mail newsletter.

To create a check box inviting customers to subscribe:

1. Return to the **survey.htm** file in your text editor.
2. Directly above the closing `</form>` tag, insert the following code (see Figure 6-45):

```
<label id="newsletter">
   <input type="checkbox" name="newscb" />
   E-mail me your newsletter for great coupons and specials!
</label>
```

6-45 **Creating a check box for the newscb field**

```
      <label for="comments">Tell us more about your experience!</label>
      <textarea name="comments" id="comments"></textarea>

   </fieldset>

   <label id="newsletter">
      <input type="checkbox" name="newscb" />
      E-mail me your newsletter for great coupons and specials!
   </label>

</form>
```

3. Save your changes to the file.

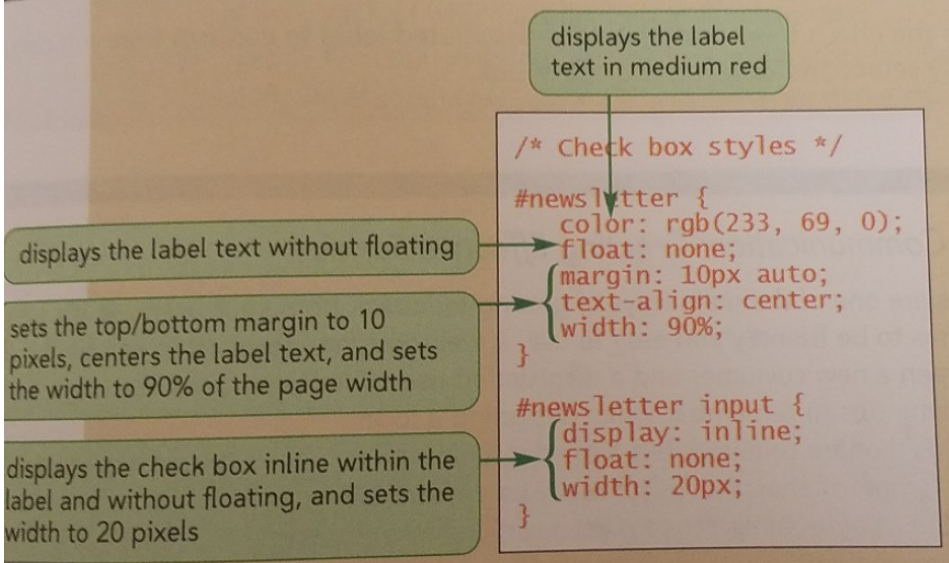
Next, you'll design a style rule for the label text and check box control.

To create a style rule for the label text and check box control:

1. Go to the **forms.css** file in your text editor.
2. At the bottom of the file, insert the following style rules (see Figure 6-46):

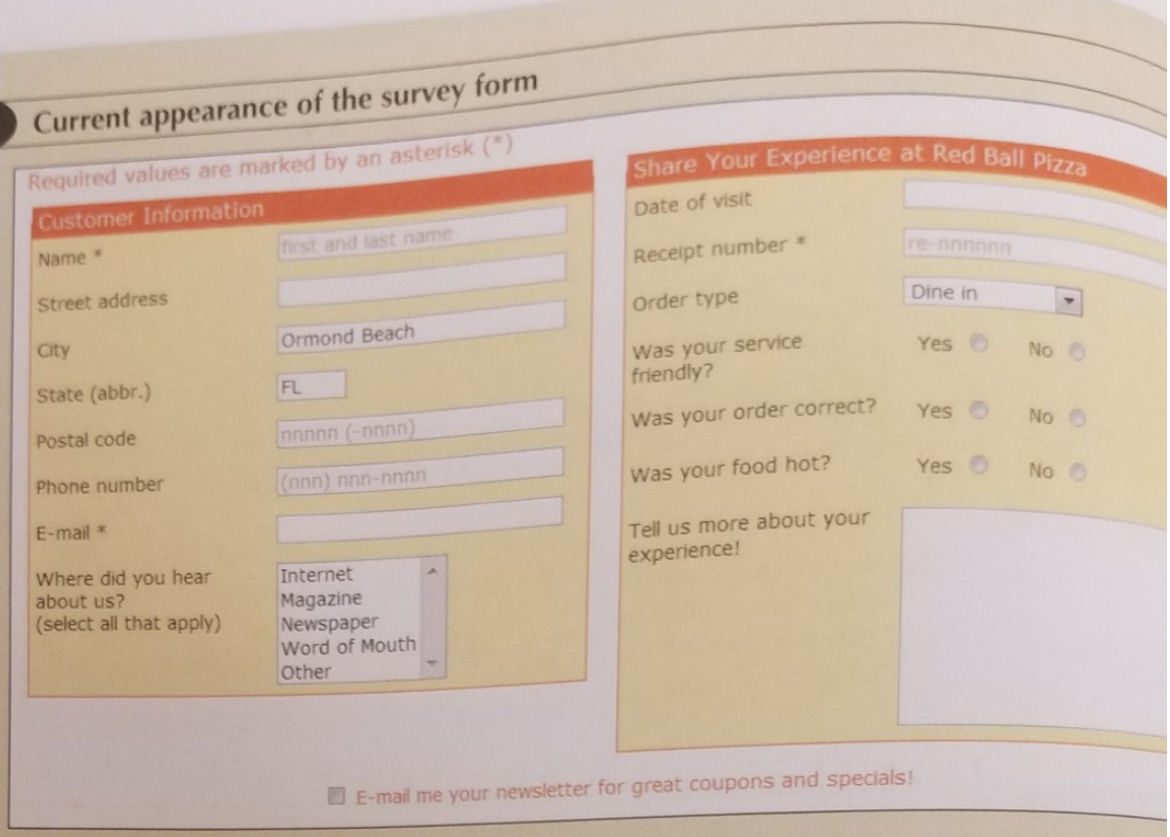
```
/* Check box styles */

#newsletter {
   color: rgb(233, 69, 0);
   float: none;
   margin: 10px auto;
   text-align: center;
   width: 90%;
}

#newsletter input {
   display: inline;
   float: none;
   width: 20px;
}
```

Style rules for the label and check box



3. Save your changes to the style sheet and then refresh **survey.htm** in your Web browser. Figure 6-47 shows the current state of the Web form.

Current appearance of the survey form



4. Click the check box control and the associated label to confirm that you can alt nately select and deselect the check box.

PROSKILLS

Written Communication: Creating Effective Forms

Web forms are one of the main ways of getting feedback from your users, so it's import for the forms to be friendly and easy to use. A well-designed form often can be the diffe ence between a new customer and a disgruntled user who leaves your site to go elsewh Here are some tips to remember when designing a form:

- Mark fields that are required, but also limit their number. Don't overwhelm your users requests for information that is not really essential. Keep your forms short and to the p
- Use the `autofocus` attribute to place users automatically into the first field of your f rather than forcing them to click that field.
- Many users will navigate through your form using the Tab key. Make sure that your ta order is logical and easy for users to follow.
- Provide detailed instructions about what users are expected to do. Don't assume that your form is self-explanatory.
- If you ask for personal data and financial information, provide clear assurances that the will be secure. If possible, provide a link to a Web page describing your security practic
- If you need to collect a lot of information, break the form into manageable sections spread out over several pages. Allow users to easily move backward and forward thro the form without losing data. Provide information to users indicating where they are they progress through your pages.
- Clearly indicate what users will receive once a form is submitted, and provide feedba on the Web site and through e-mail that tells them when their data has been success submitted.

Finally, every Web form should undergo usability testing before it is made available the general public. Weed out any mistakes and difficulties before your users see the for