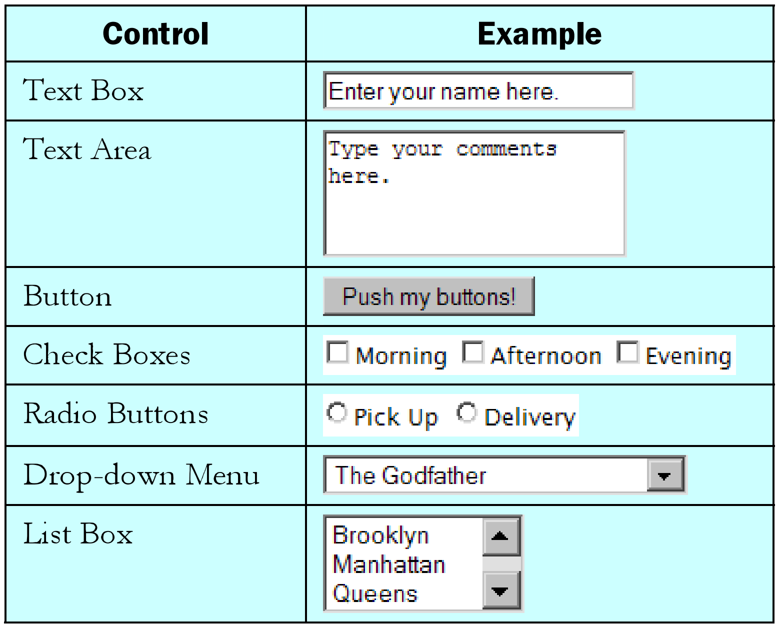
6.2 Form Controls

There are a number of HTML elements that represent user interface controls or widgets in a form. These controls allow the user to enter information that will be sent to a web server. The controls we cover in this section are buttons, radio buttons, check boxes, drop-down menus, list boxes, text boxes, and text areas. These controls are depicted in Figure 6.1.



Control	Example
Text Box	Enter your name here.
Text Area	Type your comments here.
Button	Push my buttons!
Check Boxes	☐ Morning ☐ Afternoon ☐ Evening
Radio Buttons	○ Pick Up ○ Delivery
Drop-down Menu	The Godfather ▼
List Box	Brooklyn Manhattan Queens

Figure 6.1 HTML user interface controls

In all of the examples in this section, we'll have our forms use an `action` URL of http://webster.cs.washington.edu/params.php. This is a page set up by the authors that simply echoes all query parameters that were sent to it.

6.2.1 Text Boxes (Single-Line)

Element	`input`
Description	Box for entering a single line of text (inline)
Syntax	`<input type="text" name="`**`name`**`"` **`attributes`** `/>`

The simplest kind of form input is a text box that lets the user enter text input like a street address, name, or ZIP code. HTML has two types of text entry boxes: A single-line text box and a multi-line text area.

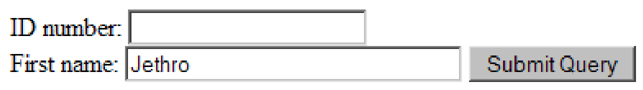
As we've already seen, a text box is created using the `input` element. Though it is not required, the tag's `type` attribute should be set to `text` to indicate a text box. We'll also give the tag a `name` attribute with a unique value.

A text box can specify additional attributes in its `input` tag. The `value` attribute specifies any initial text you want in the box. The `size` attribute designates how many characters long the text box should appear on the screen. The user can still type in more than `size` characters, but some will scroll out of view if the user types too many. The `maxlength` attribute constrains the maximum number of characters the user can enter into the text box. Table 6.2 lists these attributes. Example 6.4 demonstrates two text boxes, one with several additional attributes set.

Attribute	Value(s)	Definition and Usage
`value`	text	initial text to appear in text box
`size`	integer	visible length of text box, in characters
`maxlength`	integer	maximum number of characters that may be typed into text box

Table 6.2 Attributes for text boxes

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    ID number:
    <input type="text" name="idnumber" /> <br />
    First name:
    <input type="text" name="fname" size="30"
        maxlength="20" value="Jethro" />
    <input type="submit" />
  </div>
</form>
```



ID number:

First name: Jethro

Example 6.4 Text boxes

Recall that text boxes, like all other UI controls, are inline elements. If we want them to appear on separate lines, we must separate them with `br` tags or enclose them in separate block elements such as `div`s.

6.2.2 Text Areas (Multi-Line)

Element	`textarea`
Description	Box for entering multiple lines of text (inline)

Syntax	`<textarea name="`**name**`" rows="`**height**`" cols="`**width**`">`
	 initial text
	 `</textarea>`

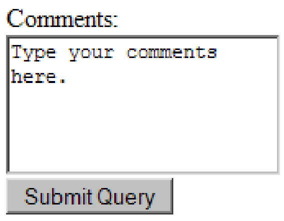
If you would like to allow the user to enter more than one line of text, use the `textarea` element. Unlike an `input` tag, `textarea` is not self-closing. Despite its multiple lines and potentially large size, a `textarea` is considered an inline element and must be enclosed in a proper block element; for example, a `textarea` cannot be placed directly inside the page's `body`.

Any text placed between the opening `<textarea>` and closing `</textarea>` tags will appear as the initial text in the area when the page loads. If no text is written, the text area will be initially blank.

The `rows` and `cols` attributes designate how many rows and columns the text area should have. These are not hard limits on the amount of text that can occupy the text area; if more text is typed, a scrollbar appears.

Example 6.5 demonstrates a text area. Notice that the closing `</textarea>` tag is not indented. This is because if you do indent it, that whitespace becomes part of the initial text inside the text area. If you don't want this, you must not indent the closing tag.

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    Comments: <br />
    <textarea name="comments" rows="4" cols="20">
Type your comments here.
</textarea> <br />
    <input type="submit" />
  </div>
</form>
```



Example 6.5 Multi-line text area

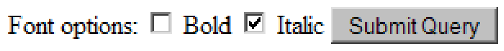
By default the words in a `textarea` wrap to the next line if it becomes too long. There is not actually any standards-compliant way to turn this off; currently you must use the non-standard `wrap="off"` attribute in the `textarea` tag.

6.2.3 Checkboxes

Element	`input`
Description	Checkbox (inline)
Syntax	`<input type="`**`checkbox`**`" name="`**`name`**`" />` **`text`**

Checkboxes and radio buttons allow the user to choose from a fixed set of options. Checkboxes are used when the options are independent and the user might want to choose none, one, or many of them. For example, photo hosting sites use checkboxes so the user can select to view color photos, black and white, or both. A check box is represented as an `input` tag with a `type` of `checkbox`. Example 6.6 demonstrates the use of check boxes.

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    Font options:
    <input type="checkbox" name="bold" /> Bold
    <input type="checkbox" name="italic" checked="checked" /> Italic
    <input type="submit" />
  </div>
</form>
```



Example 6.6 Check boxes

It may seem strange that so many different controls use the same `input` element in HTML. Why isn't there a `checkbox` element? It seems like the designers of forms (which were introduced in HTML 3 in 1996) didn't anticipate how widely they'd be used in the modern web, which is filled with interactive sites. Now we're stuck with these strange design decisions whenever we want to put controls on our page.

When a form with checkboxes is submitted, the browser finds all checkboxes that are checked, and submits them with a value of `on`. Checkboxes are not checked are not included in the URL. For example, if the user has the Bold box checked and the Italic box unchecked and submits the form, the following URL and query string will be fetched:

```
http://webster.cs.washington.edu/params.php?bold=on
```

If both boxes are checked when the form is submitted, the query string would be the following:

```
http://webster.cs.washington.edu/params.php?bold=on&italic=on
```

The optional `checked` attribute specifies that a checkbox should be checked initially when the page appears. If you do not set the checked attribute, the box will be initially unchecked. Oddly, the only valid value for the `checked` attribute is `"checked"`. In fact, there's no corresponding `"unchecked"` value or any way to specify an unchecked box. To be specific, if you set `checked` to have any value at all, it will check the box, but only `checked="checked"` is valid XHTML. The only way to indicate an unchecked box is by the lack of the `checked` attribute.

The reason the attribute works this way is because in past versions of HTML, you just wrote `checked`, with no equals sign or quoted value. But in XHTML every attribute needs to have a name, an equals sign, and a value in quotes. So the designers of XHTML decided that these kinds of "flag" attributes should be set to values equal to their own names if you want to enable them. We'll see other such attributes later in the chapter, such as `disabled` and `readonly`.

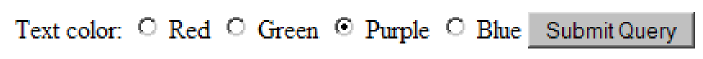
6.2.4 Radio Buttons

Element	`input`
Description	Radio button (inline)
Syntax	`<input type="radio" name="`**`group`**`" value="`**`param`**`" />` **`text`**

Radio buttons allow the user to choose one of a set of options. For example, a travel web site may use radio buttons to allow the user to select what kind of flight to search for, such as round-trip, one-way, or multi-city. A radio button is represented as an `input` tag with a `type` of `radio`.

The `name` attribute designates a group of buttons. Normally every control should have a unique name, but if you give several radio buttons the same `name`, the browser will ensure that only one radio button can be checked at a given time. Example 6.7 demonstrates the use of radio buttons.

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    Text color:
    <input type="radio" name="color" value="red" /> Red
    <input type="radio" name="color" value="green" /> Green
    <input type="radio" name="color" value="purple"
        checked="checked" /> Purple
    <input type="radio" name="color" value="blue" /> Blue
    <input type="submit" />
  </div>
</form>
```



Text color: ○ Red ○ Green ● Purple ○ Blue

Example 6.7 Radio buttons

As with checkboxes, the optional `checked` attribute specifies that a radio button should be checked initially when the page appears. Since groups of radio buttons usually require exactly one choice to be selected at all times, it is usually a good idea to set a choice to be initially checked.

Common Error

Forgetting `value` on radio buttons

The `value` attribute is optional on radio buttons, but without it, the buttons are not very useful. If you don't give values for each radio button in a group, then no matter which is checked when the form is submitted, the browser will submit that parameter with a value of `on`.

Consider the code of Example 6.8. If you test the form by submitting it to the server, you'll notice that no matter which credit card you choose, the value submitted to the server for the `ccard` query parameter is simply `on`. The server won't be able to tell which credit card the user selected.

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    <input type="radio" name="ccard" /> Visa
    <input type="radio" name="ccard" /> MasterCard
    <input type="radio" name="ccard" /> Discover <br />
    <input type="submit" />
  </div>
</form>
```

Example 6.8 Common error: Radio buttons without `value` attribute

This problem is solved using the `value` attribute. A control's `value` attribute represents the value that will be sent to the server for the corresponding query parameter when the form is submitted. The new version of the form shown in Example 6.9 submits the credit card data successfully.

```
<form action="http://webster.cs.washington.edu/params.php">
  <div>
    <input type="radio" name="ccard" value="visa" /> Visa
    <input type="radio" name="ccard" value="mastercard" /> MasterCard
    <input type="radio" name="ccard" value="discover" /> Discover
    <br /> <input type="submit" />
  </div>
</form>
```

Example 6.9 Corrected error with `value` attributes

Notice that the value doesn't need to exactly match the text next to the radio button. It can be anything you like; whatever value you need to pass to the server.

6.2.5 Labels

Element	`label`
Description	Clickable text label for a control (inline)
Syntax 1	`<label>`**control and label text**`</label>`
Syntax 2	`<label for="`**id of control**`">`**label text**`</label>`

Radio buttons and check boxes usually come with a bit of text that describes the choice, as in the examples above. However, if you try clicking on that text, you'll notice that it doesn't cause the neighboring box or button to become checked. That's because the browser doesn't have any way of knowing that the text is connected to the button. If you want the browser to associate a piece of text