Tutorial: Login Slider

In this tutorial, you'll get a little practice with using jQuery effects by creating a common user interface element: a panel that slides into and out of view with a click of the mouse (see Figure 6-2).

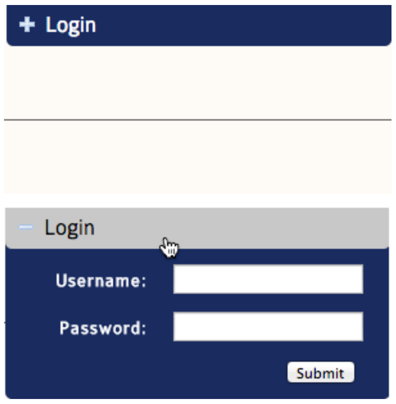

FIGURE 6-2

Now you don't see it, now you do. The login form is normally hidden from view (top), but a simple mouse click reveals the form, ready for the visitor to fill out and log on.

The basic task is rather simple:

1. **Select the paragraph with the "Login" message on it.**

   Remember that a lot of jQuery programming first begins with selecting an element on the page. In this case, the "Login" paragraph will receive clicks from a visitor.

2. **Attach a `click` event handler to that paragraph.**

   JavaScript isn't interactive without events: The visitor needs to interact with the selection (the Login paragraph) to make something happen.

3. **Toggle the display of the form on and off.**

   The previous two steps are just review (but necessary for so much of jQuery programming). This last step is where you'll use the effects you've learned about. You can make the form instantly appear (the `show()` function), slide into view (the `slideDown()` function), or fade into view (the `fadeIn()` function.)

NOTE See the note on page 12 for information on how to download the tutorial files.

The Programming

1. **In a text editor, open the file *login.html* in the *chapter06* folder.**

   This file already contains a link to the jQuery file, and the `$(document).ready()` function (page 160) is in place. First, you'll select the paragraph with the "Login" text.

2. **Click in the empty line after the `$(document).ready()` function, and then type `$('#open')`.**

   The "Login" text is inside a paragraph tag that's surrounded by a link: `<a href="form.html"><p id="open">Login</p></a>`. The link will direct users to another page that has the login form in it. The link is there so if a visitor doesn't have JavaScript turned on in her browser, she won't be able to see the hidden login form. By adding a link, you assure that even people without JavaScript turned on have an alternative way to reach a login form.

   The `$('#open')` you just typed selects the paragraph. Now, it's time to add an event handler.

NOTE The `<p>` tag mentioned in step 2 is wrapped in an `<a>` tag. If you've been building websites for a while, you may think that's invalid HTML. And it was—in HTML 4 and earlier. When you use the HTML5 doctype (page xvii), it's valid to wrap block-level elements like `<p>`, `<h1>`, and even `<div>` tags with links. Doing so is an easy way to create a large, clickable link target.

3. **Add the bolded code below, so the script looks like this:**

   ```
   $(document).ready(function() {
     $('#open').click(function(evt) {

     }); // end click
   }); // end ready
   ```

   This code adds a click handler, so each time a visitor clicks on the paragraph, something happens. In this case, the form should appear when clicked once and then disappear when clicked again, appear on the next click, and so on. In other words, the form toggles between visible and invisible. jQuery offers three functions that will serve this purpose: `toggle()`, `fadeToggle()`, and `slideToggle()`. The difference is merely in how the effect looks.

   You're also adding an argument to the anonymous function inside the `click()` method. As discussed on page 164, every event automatically gets passed an event object, which contains different properties and methods. You'll need this object to tell jQuery to stop the browser from following the link to the form page.

4. **Click in the empty line inside the `click()` function and type:**

   ```
   evt.preventDefault();
   ```

   The `preventDefault()` method stops the browser from following the link that surrounds the paragraph. Remember, the link is there to give visitors whose browsers don't have JavaScript turned on a way to reach a login form. But for browsers with JavaScript, you must tell the browser to stay on the page and run some more JavaScript.

5. **After the last line you added, type:**

```
$('#login form').slideToggle(300);
```

   This code selects the login form, slides it into view if it currently isn't visible, and then slides it back out of view. Finally, you'll change the class of the paragraph, so that the "Login" paragraph can change appearance using a CSS class style.

6. **Add the code in bold below, so the finished script looks like this:**

```
$(document).ready(function() {
  $('#open').click(function(evt) {
    evt.preventDefault();
    $('#login form').slideToggle(300);
    $(this).toggleClass('close');
  }); // end click
}); // end ready
```

   As you'll recall from page 159, when you're inside of an event handler, you can use `$(this)` to refer to the element that responds to the event. In this case, `$(this)` refers to the paragraph the visitor clicks on—the `$('#open')` in line 2 above. The `toggleClass()` function simply adds or removes a class from the element. Like the other toggle functions, `toggleClass()` adds the specified class if it's missing or removes the class if it's present. In this example, there's a class style—`.close`—in a style sheet on the page. This style simply adds a different background image to indicate that the visitor can close the login slider. (Look in the <head> and you can see the style and what it does.)

7. **Save the page and preview it in a web browser.**

   Make sure you click the "Login" paragraph several times to see how it works. You'll find a finished version of the tutorial—*complete_login.html*—in the *chapter06* folder. Try out the other toggle effects as well, by replacing `slideToggle()` with `toggle()` or `fadeToggle()`.

But what if you want two different effects? One for making the form appear—slide the form down into view, for example—and a different effect to make it disappear—fade out of view, for example. The code in step 5 won't work because the `click()` function doesn't let you choose between two different actions. You'll need to use the same trick you used in the FAQ tutorial in the previous chapter (page 174): when the visitor clicks the "Login" link, you need to check if the form is hidden. If it is, then show it; if it's visible then hide it.

To make the form slide into view then fade out of view on alternating clicks, you use this code:

```
$(document).ready(function() {
  $('#open').click(function(evt) {
    evt.preventDefault();
    if ($('#login form').is(':hidden')) {
      $('#login form').fadeIn(300);
      $(this).addClass('close');
    } else {
      $('#login form').slideUp(600);
      $(this).removeClass('close');
    }
  }); // end click
}); // end ready
```

NOTE You'll find the above code in the *completed_login2.html* file in this chapter's tutorial folder.

Animations

You aren't limited to just the built-in effects jQuery supplies. Using the `animate()` function, you can animate any CSS property that accepts numeric values such as pixel, em, or percentage values. For example, you can animate the size of text, the position of an element on a page, the opacity of an object, or the width of a border.

NOTE jQuery, by itself, can't animate color—for example, the color of text, background color, or border color. However, jQuery UI has many additional animation effects including the ability to animate color. You'll learn about jQuery UI's animation tools in Chapter 12.

To use this function, you must pass an object (page 136) containing a list of CSS properties you wish to change and the values you wish to animate to. For example, say you want to animate an element by moving it 650 pixels from the left edge of the page, changing its opacity to 50%, and enlarging its font size to 24 pixels. The following code creates an object with those properties and values:

```
{
  left: '650px',
  opacity: .5,
  fontSize: '24px'
}
```

Note that you only have to put the value in quotes if it includes a measurement like px, em, or %. In this example, you need quotes around `'650px'` because it contains px, but not around the opacity value because .5 is simply a number and doesn't contain any letters or other characters. Likewise, putting quotes around the property (`left`, `opacity`, and `fontSize`) is optional.

NOTE JavaScript doesn't accept hyphens for CSS properties. For example, `font-size` is a valid CSS property, but you can't use it in the name of key in an object literal. The hyphen has a special meaning in JavaScript (it's the minus operator). When using CSS properties as a key in an object literal, remove the hyphen and capitalize the first letter of the word following the hyphen. For example, `font-size` becomes `fontSize`, and `border-left-width` becomes `borderLeftWidth`.