In the head section of Example 4-1, after the `validator.js` file has been inserted into the program, some CSS has been included that hides (`display:none`) a division (`div`) tag with an ID of `JS`. Just below the CSS code is a small amount of JavaScript code. This code switches the `JS` division to be visible (`$("#JS").show()`) and hides the `noJS` division. There is only one slight difference between the forms in the `JS` and `noJS` divisions. You should notice that the `noJS` form does not call the `validate_input` JavaScript function.

Why is this?

The answer is simple. If the browser does not have JavaScript enabled, the short JavaScript routine to hide and show the divisions will not run. Thus, the JS division will not be displayed in the browser (because the CSS at the top of the code set its display to none). The `noJS` division will be visible instead. This allows a browser that does not have JavaScript to send the information entered by the user directly to the PHP program on the web server. Hopefully, the user has an HTML5 browser that will still validate the information before it is sent to the server (#1 in this scenario). However, even if they don't, the information will be validated on the server (#3 in this scenario, as you will see later in the chapter.

If JavaScript is enabled in the browser, the hide-show routine will "hide" the `noJS` division and show the JS division. The form that submits the information to the `validate_input` function will display in the browser. This will allow browsers that have JavaScript enabled to validate information entered by using the `validate_input` JavaScript function. If the user has an HTML5 browser (#1 in this scenario), this might be overkill, since HTML5 can validate the information. However, you must be prepared for users who have not upgraded their browsers (#2 in this scenario).

Figure 4-1. *The lab.html file*

In Example 4-1, the `dog_name`, `dog_breed`, and `dog_color` text boxes set field lengths to the same maximum values that were validated in the `Dog` class in Chapter 3. They also use the HTML5 `pattern` property ([a-zA-Z]) to only allow alphabetic characters. The `title` property is used to display an error to the user if incorrect information is entered. The `dog_weight` text box uses an input type of `number`, which automatically restricts the input. The `min` and `max` parameters are also set to restrict the dog's weight between 1 and 120 lbs. As mentioned, the browser might not interpret these restrictions if it has not fully implemented HTML5 standards. Therefore, you still must also validate the data using the JavaScript `validate_input` method to be sure that all validation has been completed.