```
        else
        {
        return true;
        }
}

<script src="validator.js"></script>
```

In Example 4-1, the JavaScript file `validator.js` is attached to the `lab.html` file in the head section via an HTML `script` tag. By placing the JavaScript validation in a separate file, you have simplified the HTML file. The JavaScript file is dependent on the HTML file for display. Error messages are passed to an alert box (`alert(error_message);`) near the bottom of the method.

> *Programming suggestion—By passing only a status (error message) back to a calling program, you allow for flexibility. The calling program can then determine how to display the information. This also allows you to easily reuse the method for other HTML forms.*

Let's work backward in reviewing the JavaScript code (Example 4-2). In the HTML file, the `onSubmit` parameter of the `form` tag calls the `validate_input` JavaScript method. This method controls all the validation of each text box on the form. The method uses a series of `if` statements to call methods that will validate the different types of text boxes. The format of each `if` statement is very similar.

```
if (!validate_dog_name(form.dog_name.value))
```

Each method called will return a true response if the validation passes and a false response if the validation fails. (In JavaScript, the constants `true` and `false` are lowercase. In PHP they are uppercase.)

By **passing** `form` into the `validate_input` function, all the parameters and values in the HTML form are passed at the same time. You can select which parameter to use by using dot notation.

In the previous example, `form.dog_name.value` will pull the value that was entered in the `dog_name` text box. The value that the user entered in this text box is passed into the `validate_dog_name` method.

```
if (!validate_dog_name(form.dog_name.value))
{
    error_message += "Invalid dog name. ";
}
```

The `!` symbol in front of the function name means `not`. An `if` statement normally executes if the response from a method (or comparison) is `true`. However, you want to save an error message in the `error_message` property if the response returned from the method (`validate_dog_name`) is `false` (it did not validated correctly). The `!` symbol causes the `if` statement to do the reverse of what's normal and execute if the function returns a false value.

Each of the other validation methods works in similar ways. The methods check to see if the information entered was in the correct format. If the information was correct, the method returns `true`. If the information was not correct, the method returns `false`.

> *For more information on the JavaScript if conditional statement, visit*
>
> *https://www.thenewboston.com/videos.php?cat=10&video=16960*
>
> *For more information on JavaScript functions (methods), visit*
>
> *https://www.thenewboston.com/videos.php?cat=10&video=16952*