Secured User Interaction

In Chapter 2, the `Hello World` examples included user interaction (clicking a Submit button) to call a PHP program. This chapter will use the HTML web form to accept information from the user and then pass this information to a PHP program.

Do not trust your users! You must be prepared for any type of information your users will enter into a web form. You must also make sure that you validate and secure information every time before accepting it into your programs. You must remember that users may elect to not allow JavaScript to run in their browsers. Therefore, your program cannot be dependent on JavaScript code for verification of input. You must be able to handle these three scenarios:

1. If the user is using an HTML5 capable browser, you can verify all input using HTML5 before sending it to the PHP program on the web server.
2. If the user is using a browser that does not have HTML5 (or complete HTML5) capability, you can verify all (or some) input using JavaScript. Then you can send the validated information to the PHP program on the web server.
3. If the user is using a browser that does not have HTML5 capability and has JavaScript disabled, the user input can still be verified by the PHP program on the web server.

It is preferable to handle the initial verification using either of the first two methods. This would verify the correct information before sending it to the server. Method #3 will cause more server calls because the information must be sent to the server for verification, and then any error messages must be sent back to the browser for the user to correct.

Even if you verify the information using method #1 or #2, when the PHP programs receives the information on the web server, you will again evaluate that you have received valid information. Even though the user may have sent valid information, packet sniffing programs can change valid information into harmful information.

HTML5 Form Validation

When building an HTML5 form, you can validate information entered into text boxes. However, browsers that have not implemented the HTML5 techniques (or all of HTML5) will treat the objects (such as text boxes) as if they were normal non-validation objects. The information in the boxes will be accepted without any verification. Therefore, you must also include a JavaScript routine to catch anything not validated by HTML5.

> *Security and performance —Secure programming is just part of the complete process of protecting your information. The files, directories, servers, networks, and databases must also be properly secured. In addition, any highly important information (such as credit card numbers) must be sent across a secure channel (HTTPS) to additionally protect the information. User IDs and passwords should be encrypted to make it difficult for packet sniffing software.*

You will continue with the example from Chapter 3 by updating the `Dog` class example to accept information from the user for your properties. The `Dog` class already has security in place, so you will not need to make any additional security updates to the class.