1

PHP Crash Course

THIS CHAPTER GIVES YOU A QUICK OVERVIEW of PHP syntax and language constructs. If you are already a PHP programmer, it might fill some gaps in your knowledge. If you have a background using C, Perl Active Server Pages (ASP), or another programming language, it will help you get up to speed quickly.

In this book, you'll learn how to use PHP by working through lots of real-world examples taken from our experiences building real websites. Often, programming textbooks teach basic syntax with very simple examples. We have chosen not to do that. We recognize that what you do is to get something up and running, and understand how the language is used, instead of plowing through yet another syntax and function reference that's no better than the online manual.

Try the examples. Type them in or load them from the CD-ROM, change them, break them, and learn how to fix them again.

This chapter begins with the example of an online product order form to show how variables, operators, and expressions are used in PHP. It also covers variable types and operator precedence. You learn how to access form variables and manipulate them by working out the total and tax on a customer order.

You then develop the online order form example by using a PHP script to validate the input data. You examine the concept of Boolean values and look at examples using `if`, `else`, the `?:` operator, and the `switch` statement. Finally, you explore looping by writing some PHP to generate repetitive HTML tables.

Key topics you learn in this chapter include

- Embedding PHP in HTML
- Adding dynamic content
- Accessing form variables

- Understanding identifiers
- Creating user-declared variables
- Examining variable types
- Assigning values to variables
- Declaring and using constants
- Understanding variable scope
- Understanding operators and precedence
- Evaluating expressions
- Using variable functions
- Making decisions with `if`, `else`, and `switch`
- Taking advantage of iteration using `while`, `do`, and `for` loops

Before You Begin: Accessing PHP

To work through the examples in this chapter and the rest of the book, you need access to a web server with PHP installed. To gain the most from the examples and case studies, you should run them and try changing them. To do this, you need a testbed where you can experiment.

If PHP is not installed on your machine, you need to begin by installing it or having your system administrator install it for you. You can find instructions for doing so in Appendix A, "Installing PHP and MySQL." Everything you need to install PHP under Unix or Windows can be found on the accompanying CD-ROM.

Creating a Sample Application: Bob's Auto Parts

One of the most common applications of any server-side scripting language is processing HTML forms. You'll start learning PHP by implementing an order form for Bob's Auto Parts, a fictional spare parts company. You can find all the code for the examples used in this chapter in the directory called `chapter01` on the CD-ROM.

Creating the Order Form

Bob's HTML programmer has set up an order form for the parts that Bob sells. This relatively simple order form, shown in Figure 1.1, is similar to many you have probably seen while surfing. Bob would like to be able to know what his customers ordered, work out the total prices of their orders, and determine how much sales tax is payable on the orders.

Figure 1.1 Bob's initial order form records only products and quantities.

Part of the HTML for this form is shown in Listing 1.1.

Listing 1.1 `orderform.html`**— HTML for Bob's Basic Order Form**

```
<form action="processorder.php" method="post">
<table border="0">
<tr bgcolor="#cccccc">
  <td width="150">Item</td>
  <td width="15">Quantity</td>
</tr>
<tr>
  <td>Tires</td>
  <td align="center"><input type="text" name="tireqty" size="3"
     maxlength="3" /></td>
</tr>
<tr>
  <td>Oil</td>
  <td align="center"><input type="text" name="oilqty" size="3"
     maxlength="3" /></td>
</tr>
```

Listing 1.1 **Continued**

```
<tr>
  <td>Spark Plugs</td>
  <td align="center"><input type="text" name="sparkqty" size="3"
     maxlength="3" /></td>
</tr>
<tr>
  <td colspan="2" align="center"><input type="submit" value="Submit Order" /></td>
</tr>
</table>
</form>
```

Notice that the form's action is set to the name of the PHP script that will process the customer's order. (You'll write this script next.) In general, the value of the `action` attribute is the URL that will be loaded when the user clicks the Submit button. The data the user has typed in the form will be sent to this URL via the method specified in the `method` attribute, either `get` (appended to the end of the URL) or `post` (sent as a separate message).

Also note the names of the form fields: `tireqty`, `oilqty`, and `sparkqty`. You'll use these names again in the PHP script. Because the names will be reused, it's important to give your form fields meaningful names that you can easily remember when you begin writing the PHP script. Some HTML editors generate field names like `field23` by default. They are difficult to remember. Your life as a PHP programmer will be easier if the names you use reflect the data typed into the field.

You should consider adopting a coding standard for field names so that all field names throughout your site use the same format. This way, you can more easily remember whether, for example, you abbreviated a word in a field name or put in underscores as spaces.

Processing the Form

To process the form, you need to create the script mentioned in the `action` attribute of the `form` tag called `processorder.php`. Open your text editor and create this file. Then type in the following code:

```
<html>
<head>
  <title>Bob's Auto Parts - Order Results</title>
</head>
<body>
<h1>Bob's Auto Parts</h1>
<h2>Order Results</h2>
</body>
</html>
```

Notice how everything you've typed so far is just plain HTML. It's now time to add some simple PHP code to the script.

Embedding PHP in HTML

Under the `<h2>` heading in your file, add the following lines:

```
<?php
  echo '<p>Order processed.</p>';
?>
```

Save the file and load it in your browser by filling out Bob's form and clicking the Submit Order button. You should see something similar to the output shown in Figure 1.2.

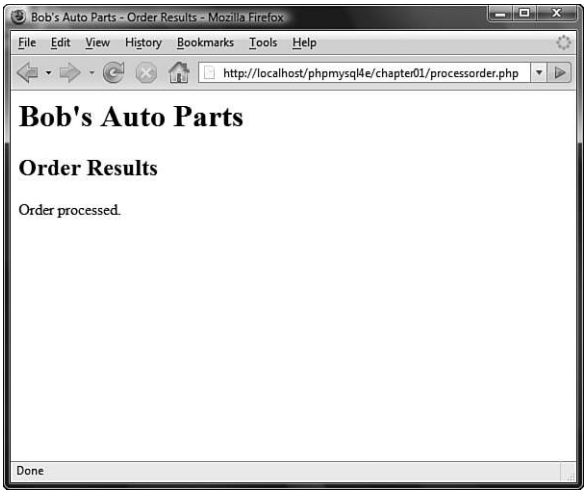

Figure 1.2 Text passed to PHP's `echo` construct is echoed to the browser.

Notice how the PHP code you wrote was embedded inside a normal-looking HTML file. Try viewing the source from your browser. You should see this code: