Some Useful Notes:

1. Always use quotation for strings if the string besides a function e.g. date(‘H:i, jS F’).
2. Embedding PHP in HTML:
3. PHP Opening and Closing Tags:

1. <?php echo 'if you want to serve PHP code in XHTML or XML documents,

use these tags'; ?>

2. <? echo 'this code is within short tags'; ?>

Code within these tags <?= 'some text' ?> is a shortcut for this code <? echo 'some text' ?>

3. <% echo 'You may optionally use ASP-style tags'; %>

Code within these tags <%= $variable; %> is a shortcut for this code <% echo $variable; %>

Note:

Using short tags should be avoided when developing applications or libraries that are meant for redistribution, or deployment on PHP servers which are not under your control, because short tags may not be supported on the target server. For portable, redistributable code, be sure not to use short tags.

Note:

<= is recognized and valid after PHP 5.4.

1. Alternative syntax for control structures :

PHP provides the alternative syntax for

* If, while, for, foreach, and switch statements.
* Opening braces ({) replaced by colon (:),
* Closing brace is replaced by an endif;, endwhile;, endfor;, endforeach;, or endswitch;, respectively.

I included an example of each below.

Example:

<?php $a=1; $b=4;?>

<p>----Example IF Statement<br>

<?php if ($a == $b): ?>

A equals B<br>

<?php else:?>

A does not equal B<br>

<?php endif; ?></p>

<p>----Example WHILE Statement<br>

<?php while($a < $b): ?>

Here is some code inside a while<br>

<?php

$a++;

endwhile; ?>

</p>

<p>----Example FOR Statement<br>

<?php for($i = 1; $i <= 10; $i++):?>

Some code in a for<br>

<?php endfor; ?>

</p>

1. In PHP we can access each form field as a PHP variable whose name relates to the name of the form field.
2. Depending on your PHP version and setup, you can access the form data via variables in three ways.

$tireqty // short style

$\_POST[‘tireqty’] // medium style

$HTTP\_POST\_VARS[‘tireqty’] // long style

* Short style ($tireqty) is convenient but requires the register\_globals configuration setting be turned on. For security reasons, this setting is turned off by default.
* Medium style ($\_POST[‘tireqty’]) is the recommended approach.
* Long style ($HTTP\_POST\_VARS[‘tireqty’]) is the most verbose. Note, however, that it is deprecated and is therefore likely to be removed in the long term.
* When you use the short style, the names of the variables in the script are the same as the names of the form fields in the HTML form.
* Medium style involves retrieving form variables from one of the arrays $\_POST, $\_GET, or $\_REQUEST. One of the $\_GET or $\_POST arrays holds the details of all the form variables.
* In addition, a combination of all data submitted via GET or POST is also available through $\_REQUEST.
* If the form was submitted via the POST method, the data entered in the tireqty box will be stored in $\_POST[‘tireqty’]. If the form was submitted via GET, the data will be in $\_GET[‘tireqty’]. In either case, the data will also be available in $\_REQUEST[‘tireqty’].
* String Concatenation helpful to avoid writing multiple echo commands.

1. The complete list of superglobals is as follows:

$GLOBALS—An array of all global variables (Like the global keyword, this allows you to access global variables inside a function—for example, as

$GLOBALS[‘myvariable’].)

$\_SERVER—An array of server environment variables

$\_GET—An array of variables passed to the script via the GET method

$\_POST—An array of variables passed to the script via the POST method

$\_COOKIE—An array of cookie variables

$\_FILES—An array of variables related to file uploads

$\_ENV—An array of environment variables

$\_REQUEST—An array of all user input including the contents of input including $\_GET, $\_POST, and $\_COOKIE (but not including $\_FILES since PHP 4.3.0)

$\_SESSION—An array of session variables

1. The Error Suppression Operator:

The error suppression operator (@) can be used in front of any expression—that is, anything that generates or has a value. For example,

$a = @(57/0);

Without the @ operator, this line generates a divide-by-zero warning.With the operator included, the error is suppressed.

1. If you have PHP set up with the track\_errors feature enabled in php.ini, the error message will be stored in the global variable $php\_errormsg.
2. The Execution Operator

The execution operator is really a pair of operators—a pair of backticks (``) in fact.The backtick is not a single quotation mark; it is usually located on the same key as the ~ (tilde) symbol on your keyboard.PHP attempts to execute whatever is contained between the backticks as a command at the server’s command line.

In Windows server, we can use

$out = `dir c:`;

echo ‘<pre>’.$out.’</pre>’;

1. The Type Operator

There is one type operator: instanceof. This operator is used in object-oriented programming, but we mention it here for completeness.

1. PHP also provides some specific type-testing functions like is\_array(),is\_double(). etc.