**PHP**

PHP is usually called Personal Home Page (PHP). It is among the most stable and efficient tools available for creating dynamic, interactive Web pages. PHP is open-source software and is used on more than 1.5 million Web sites around the world. PHP can use almost any Web server – we will be using the Apache Web server software.

**A Simple Beginning PHP Script Example: Hello World!**

Use jpico to create helloworld.php as shown below. As you can see below, PHP commands are identified to the Apache server with the special HTML <? and ?> tag combination—note that PHP 5 recommends the beginning tag include php as shown<?php code goes here ?> ; however, it will work either way. These tags are all you need to tell the Web server to process PHP code in the file. All statements in PHP are terminated with a semicolon (;) Failure to do so results in a parse error message.

<html>

<head>

<title> Hello World Example </title>

</head>

<body>

<?php

Echo "Hello, world!";

?>

<br>

<?

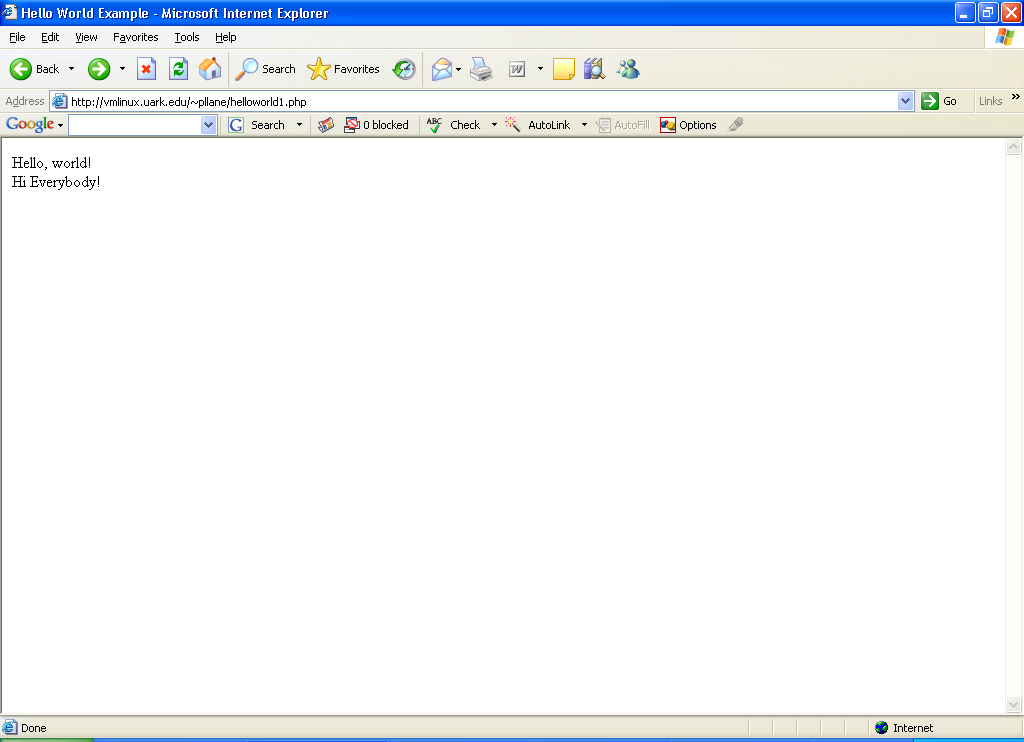
Echo "Hi Everybody!";

?>

</body>

</html>

Result:



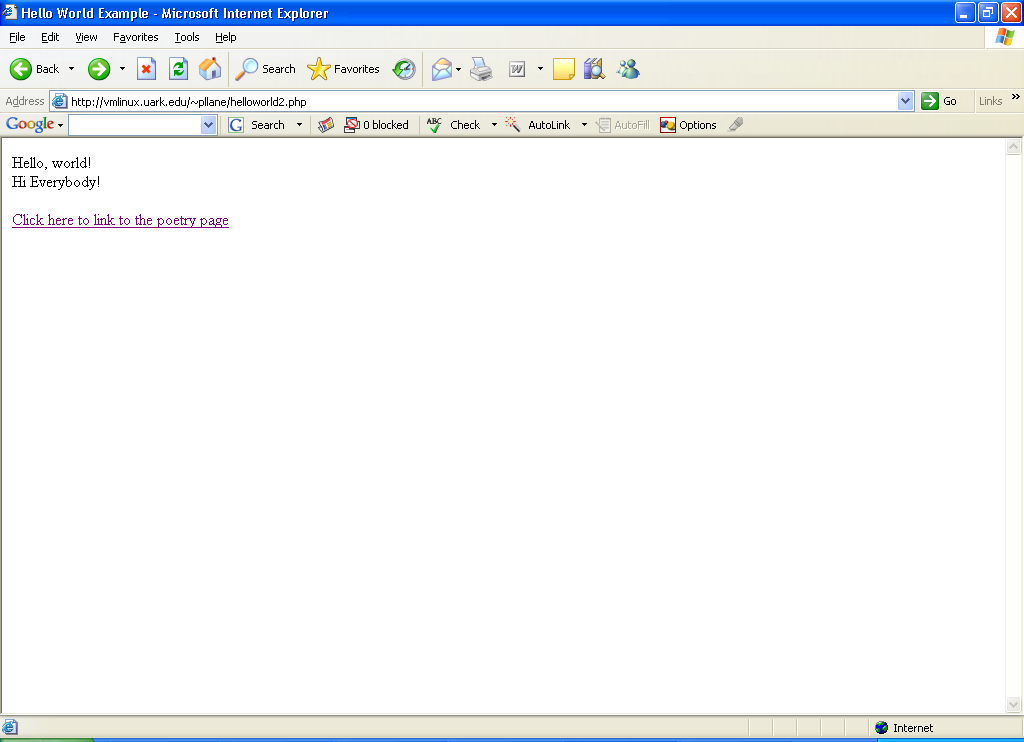
**Including Links to other web pages**

copy the helloworld.php file above to helloworld2.php (give it this exact name; do not use any uppercase letters). To helloworld2.php, add the following statement before the

</body> line.

<a href="poetry.htm"> Click here to link to the poetry page </a>

When you view helloworld2.php, you should see: Test the link. Notice the poetry page has a link back to helloworld2.php. Find the statement in poetry.htm that makes that work.



**Including Files**

When developing more than a single home page for the Internet, you probably want the pages to have a common look and feel. To make this possible, PHP has provided a method called included files. These files let you incorporate common artwork, contact information, and menu and link options into your Web pages with a minimum of code.

To include files in a PHP script:

1. Create a new document named header.inc with your editor and type the following:

<html>

<head>

<title> Contents of Header.inc are here</title>

</head>

<body>

<h1>This is a common header line </h1>

<h2> to be used across all my Web pages</h2>

2. Open a new document named mainpage.php in your editor and enter the following code:

<?php

include(”header.inc”);

?>

<br><br>

<a href=”secondpage.php”> Go to Second Page </a>

</body>

</html>

3. Open a new document named secondpage.php in your editor and enter the following code:

<?php

include(“header.inc”);

?>

<br><br>

<a href=”mainpage.php”> Return to Main page </a>

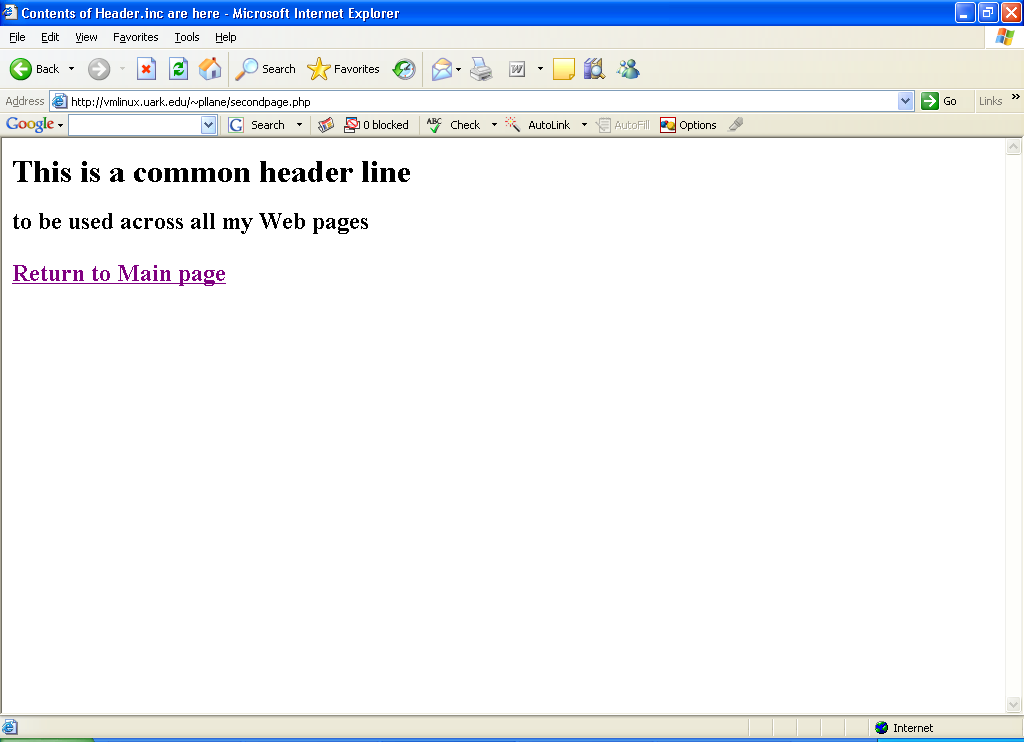
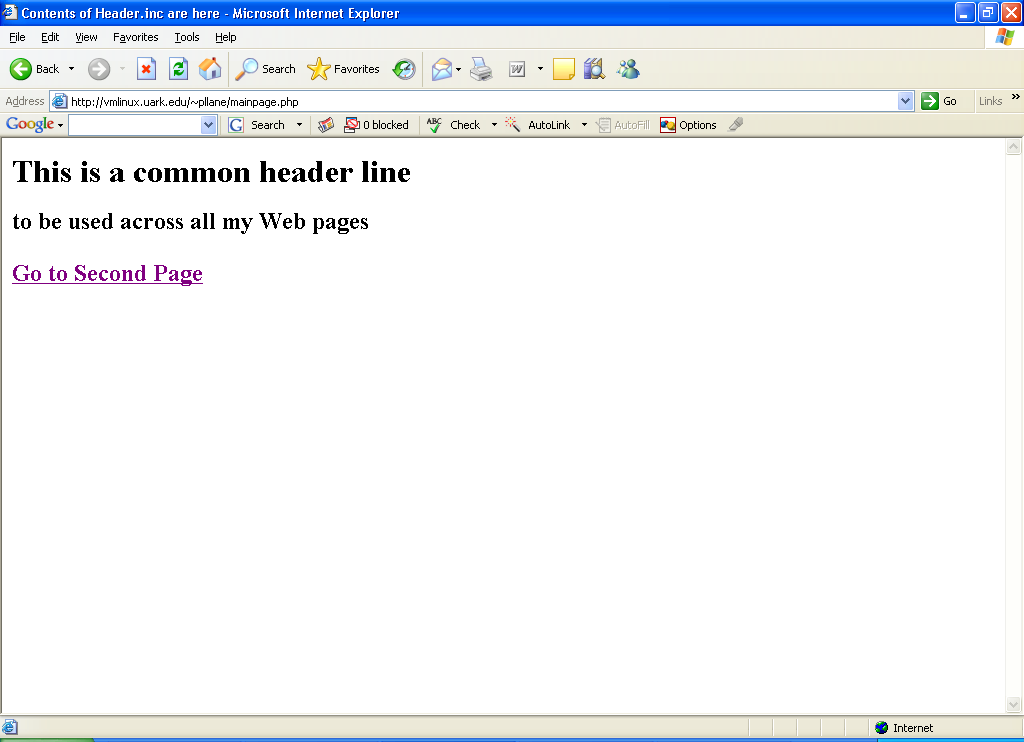
</body>

</html>

4. In Internet Explorer, open the file mainpage.php.

Go to the second page.

Return to Main Page.



Edit the header.inc file to make a real header that you could use.

So far, we’ve learned the following about PHP:

* All PHP statements end in a semicolon (;)
* Anything between the <?php And ?> tags is interpreted as PHP code and converted into HTML on the server before being sent to the requesting client machine.

**Naming Variables in PHP**

* Variables in PHP are preceded with a dollar sign ($) and contain either letters or numbers.
* Valid characters are uppercase or lowercase letters, numbers, or the underscore character.
* Do not allow the first character after the ($) to be a number.
* Variable names are case sensitive. $UserName and $username are considered different variables.
* Assign the variable an initial value with an equals (=) sign.
* If desirable, inline comments can follow the definition of a variable. The comments begin with the double slash characters: //
* Example: <?php $Password = “bigdog” //this is where the comment goes ?>

**Data Types in PHP**

Data types in PHP are Integer, Double, Boolean, and String. These variables are assigned by simply assigning a value also assigns it the data type of that value.

Examples:

<?php $Password = “bigdog”; // $Password is a string data type ?>

<?php $Number = 192; // $Number is an integer ?>

<?php $Money = 10.03; // $Money is a Double?>

**Using Operators in PHP**

|  |  |
| --- | --- |
| PHP Operator | Description |
| **Variable operators** | xyz |
| =, +, -, \*, / | Normal operators. Example: $value = $one \* $two; |
| % | Modulus (remainder) operator; $value = $one % $two; |
| **Comparison operators** | xyz |
| = = | Equality $this = = $that |
| = = = | Identical values and data type $this = = = $that |
| != | Not equal: $this !+ $that |
| <, >, <=, >= | Normal operators. Example $this <= $that |
| **Logical operators** | xyz |
| ! | NOT: ! $this Returns TRUE if $this is False |
| &&, and | AND: $this && $that Returns TRUE if both $this and $that are true |
| ||, or | OR: $this || $that Returns True if either or both $this and $that are true |
| xor | XOR: $this xor $that Returns true if one or the other true but not both |

**Using Variable Scope**

If a variable is defined at the start of a PHP file, it stays in memory until the end of that file. This is known as the variable’s scope. If a variable is assigned a value of 5 in one PHP file, and that file calls another PHP file that has a variable of the same name, then the first variable is terminated and its value is lost. One major distinction that relates to a variable’s scope involves the processing of web-based forms. Any variables that are defined within a PHP/HTML form and sent to the server with the form’s Post method are automatically sent with the called Post action and named in PHP by the same name used in the HTML form. In order for the form data to be sent with the POST method, the Register Globals must be set to ON in the php.ini file. That task is performed by someone who has access to the system. Consider the code in the example below.

Example: The file form\_proj2.htm references the file display.php to display data collected in a form. Notice that form\_proj2 does not contain any PHP code, but the file display.php does contain PHP code. Use Internet Explorer to open form\_proj2.htm, enter data, and then click submit the form. &nbsp is used to include a horizontal space.

**Code for form\_proj2.htm:**

<html>

<head>

<title> Collecting Some Information </title>

</head>

<body>

<h1> Please provide the following information: </h1>

<br><br>

**<form action="display.php" method="post">**

First name: &nbsp;

<input type=text name="firstname" size=20><br><br>

Last name: &nbsp;

<input type=text name="lastname" size=25><br><br>

Address: &nbsp;

<input type=text name="address" size=25><br><br>

City: &nbsp;

<input type=text name="city" size=15><br><br>

Telephone Number: &nbsp;

<input type=text name="phone" size=15><br><br>

<input type=submit name="submit" value="Submit form">&nbsp;&nbsp;

<input type=reset name="reset" value="Clear form">

</form>

</body>

</html>

**Code for display.php:**

<html>

<head>

<title> Display the contents of a submitted form </title>

</head>

<body>

<h1>The following data was collected: </h1>

<table><tr><td> First Name: </td>

<td><?php echo $\_POST[firstname]; ?></td></tr>

<tr><td> Last Name: </td>

<td><?php echo $\_POST[lastname]; ?></td></tr>

<tr><td> Address: </td>

<td><?php echo $\_POST[address]; ?></td></tr>

<tr><td> City: </td>

<td><?php echo $\_POST[city]; ?></td></tr>

<tr><td> Telephone Number: </td>

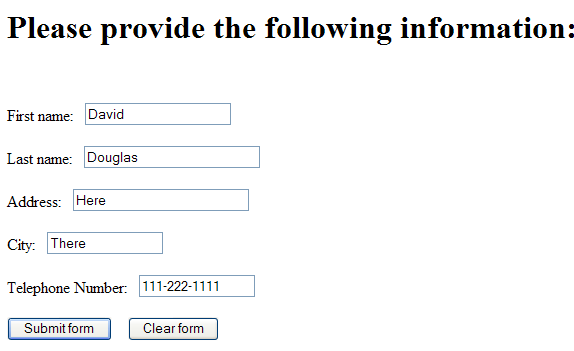
<td><?php echo $\_POST[phone]; ?></td></tr>

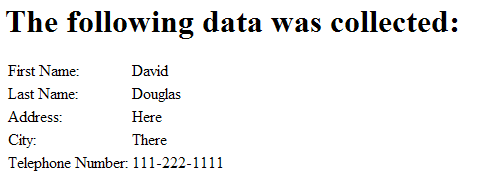
</table>

</body>

</html>

**Results:**

****



Change both programs – one to accept the zip code before telephone number and the other to display the zip code before telephone number

Note: If any of the fields turn yellow, it is because Windows is flagging those fields that can be autofilled.

**Tax calculation example** (taxcalc.php). This program is one complete PHP program. It contains the code to accept the data (amount and rate), the code to calculate the tax amount, and also the code to post the data. Notice the format statements for amount and the calculated tax that format the result to two decimal places.

<html>

<head>

<title> Sales Tax Calculator </title>

</head>

<?php

If ($\_POST['SecondTime'] == "") {

echo"

<body><h1>Please enter amount and tax rate rate</h1>

<br><br>

<form action="taxcalc.php" method="post">

<input type="hidden" name="SecondTime" Value=1>

Amount &nbsp;

<input type=text name="amount" size=15><br><br>

Tax Rate: &nbsp;

<input type=text name="rate" size=15> % <br><br>

<input type=submit name="submit" value="Submit form">&nbsp;&nbsp;

<input type=reset name="reset" value="Clear form">

";

} else {

$calc\_tax = $\_POST['amount'] + ($\_POST['amount'] \* ($\_POST['rate']/100) );

$amount = number\_format($\_POST['amount'],2,'.','');

$calc\_tax = number\_format($calc\_tax,2,'.','');

$rate = $\_POST['rate'];

echo "

<body><h1>The calculated tax rate is as follows: </h1>

<table><tr><td> Entered Amouunt: </td>

<td>$ $amount </td></tr>

<tr><td> Entered Tax Rate: </td>

<td> $rate % </td></tr>

<tr><td> Calculated Tax Amount: </td>

<td>$ $calc\_tax</td></tr>

</table><br>

<form action="taxcalc.php" method="post">

<input type="hidden" name="SecondTime" Value="">

<input type=submit name="submit" value="Return to Calculator\">

</form>

";

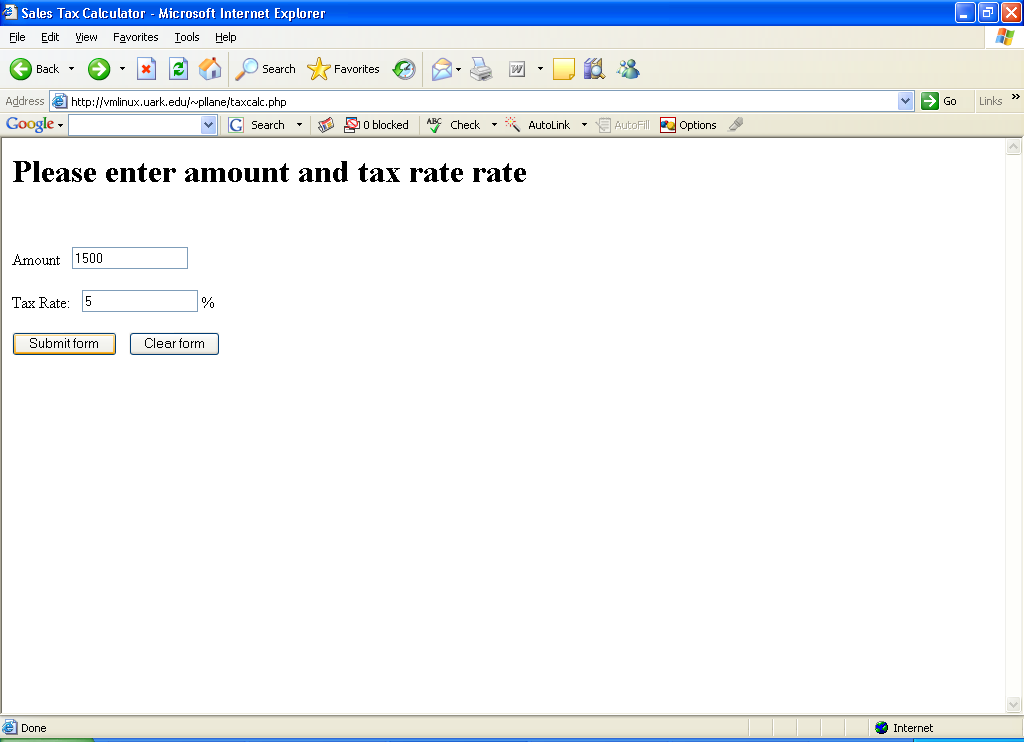
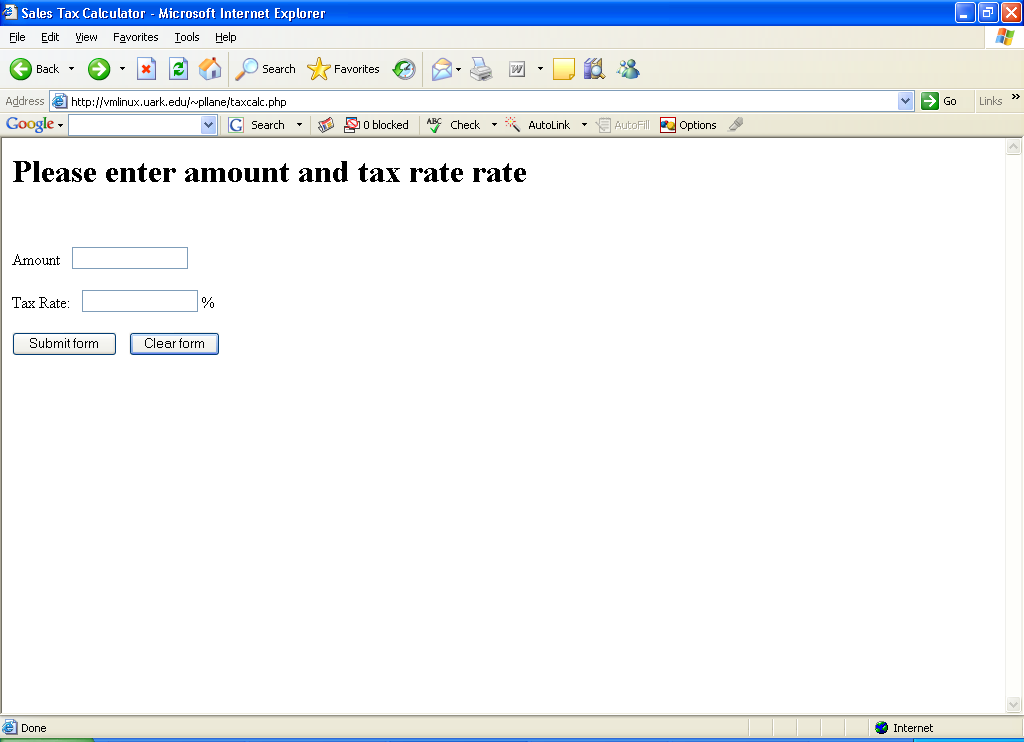
}

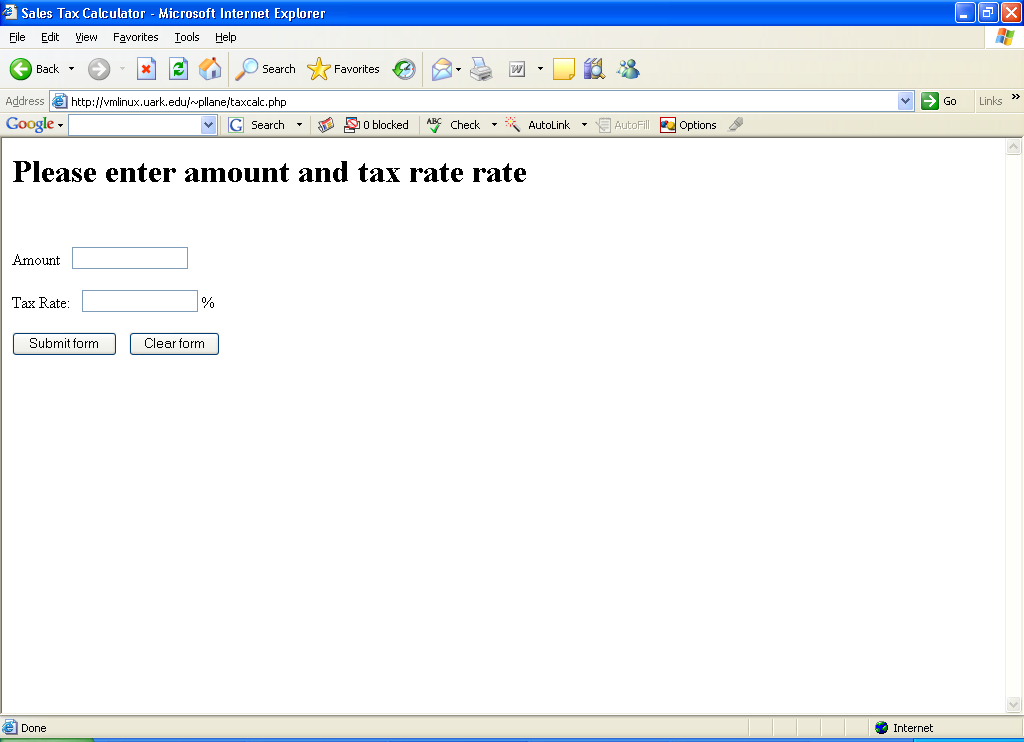
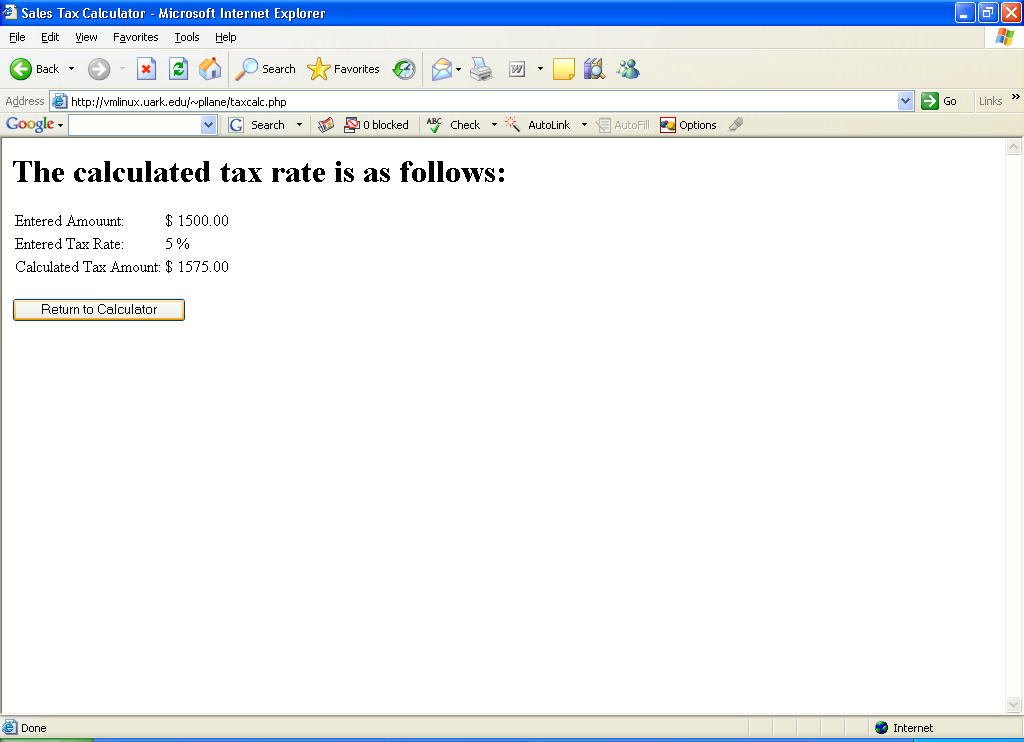
?>

</body>

</html>

Results of displaying and using taxcalc.php:





Click Return to Calculator.

**Displaying PHP Output with Print**

So far, we’ve used echo to display output. PHP has another function that allows display – Print. The print function returns a 1 or 0 integer (denoting success or failure, respectively), for the contents of the function being displayed.

Echo has a shortcut syntax, in which you can immediately follow the opening tag with an equal sign, as in <?php = $this ?>

Beyond the scope of this class, PHP also has printf() and sprintf(). Printf() displays variables by changing integers to display as decimals. Sprintf() returns a string value to PHP for future use in a program if desired.

Example of how echo and print are used in PHP is shown in output.php:

Open **output.php**in Internet Explorer:

<?php $var = “this is a string”;

echo “Anything between these quotation marks will be displayed”;

echo “ in the browser, <br> including the contents of variables: <br>

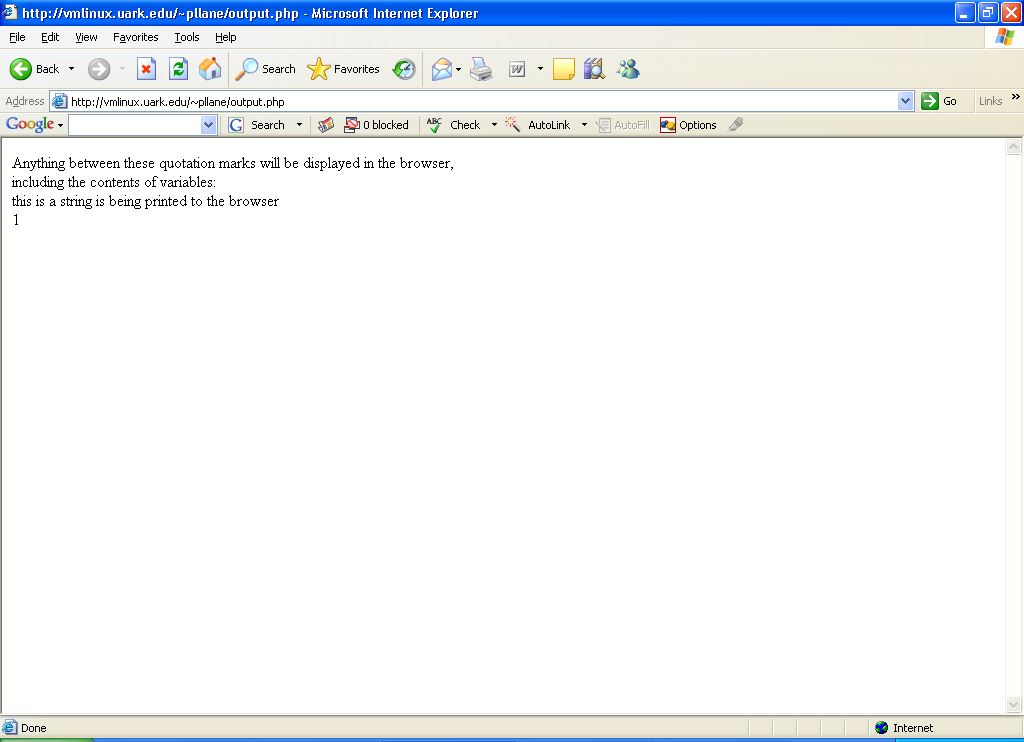
$var “;

$good = print(“ is being printed to the browser”);

echo “<br> $good”; // places 1 on the browser if previous command worked

?>

Results:



As we saw in the taxcalc.php example, if you want to send PHP reserved characters such as double quotations to the Web browser within the echo command, you must use the backslash character.

Example: echo “<br><form action=\"taxcalc.php\" method=\"post\"> “;

**Managing PHP Program Flow**

There are four construct used to manage the flow of a PHP program:

* If-then-else
* Switch-case
* For-next
* Do-while

**If-then-else**

Simple syntax:

if (condition)

{

//code based on TRUE result of condition

} else{

// code based on FALSE results of condition

}

**simpleifExample:**

<?php

if ($totalsales == 0)

{

echo “There were no sales recorded at this time …”;

} else {

echo “<h2> Thank you for shopping at our online store ! </h2>”;

echo “<h3> Please come by again soon… <br>”;

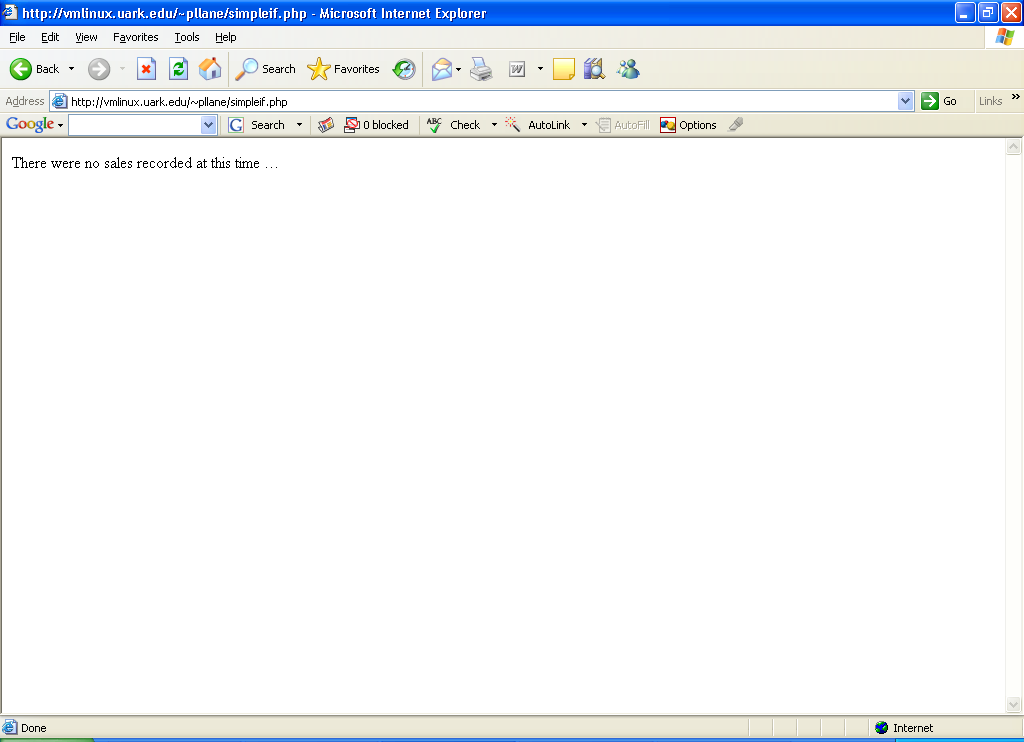
echo “Your total purchase today was: $totalsales</h3>”;

}

?>

What is printed if the value of $totalsales is 0?

How would you change the code to print There were not sales recorded at this time for zero or for negative values of $totalsales?



What does the following nested code print if $totalsales is 1000? What if it is 1500?

**nestedif.php**

<?php

if ($totalsales<= 0)

{

echo “There were no sales recorded at this time …”;

} else {

if ($totalsales> 1000)

{

echo “ <h2> Thank you for shopping at our online store! </h2>”;

echo “<h3> Please come by again soon… <br>”;

echo “Because your total purchase today was: $totalsales</h3>”;

echo “<h1> We want to give you a 10% discount on your next in-store

purchase with us ! </h1>”;

echo “<h2> Please print this page for your records, and present it at your next visit to our downtown location </h2>”;

} else {

echo “<h2> Thank you for shopping at our online store ! </h2>”;

echo “<h3> Please come by again soon… <br>”;

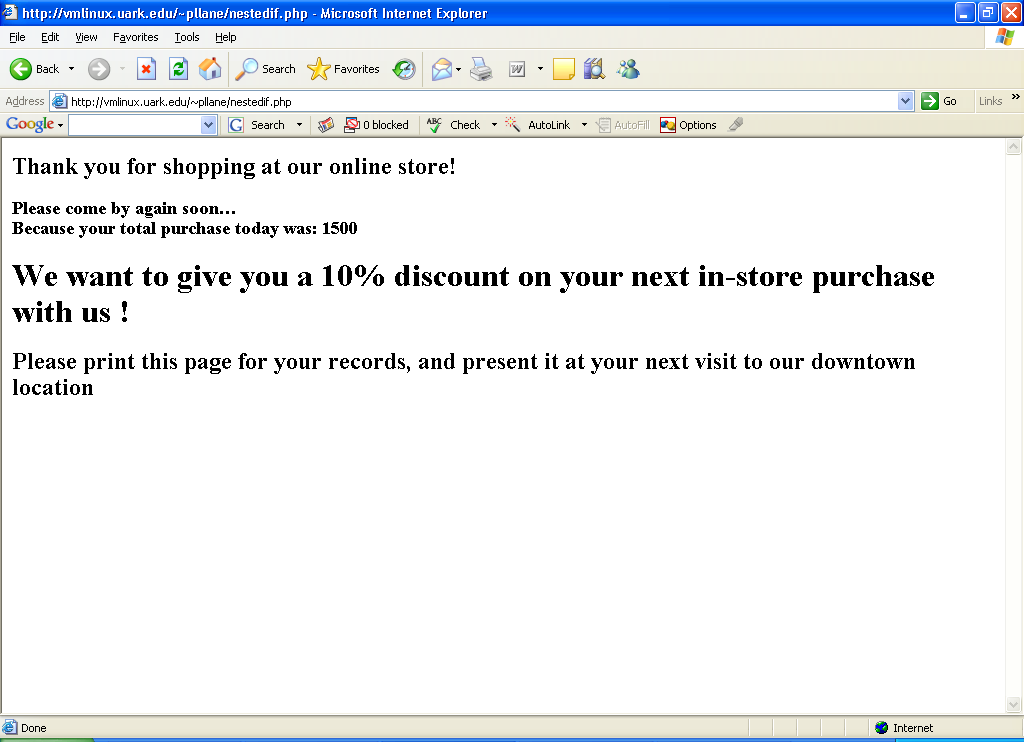
echo “Your total purchase today was: $totalsales</h3>”;

}

}

?>

Add a line of PHP code assigning the value of $totalsales to 1500 and look at the page again. You should see:



**Switch-case**

Instead of nesting multiple If statements, the Switch-case construct can be used.

Simple syntax:

switch ($value)

{

case 1 :

// code based on $value being equal to 1

[optional] break;

case 2 :

// code based on $value being equal to 2

[optional] break;

case 999 :

// code based on $value being equal to whatever number you want

[optional] break;

default :

// code based on $value being not already dealt with.

[optional] break;

}

**If the break statement is not used in each of the cases, then the cases that follow will also have their code executed.**

**Discountcase.php example:** Suppose you want to give different percentage discounts based on the amount of money that customers have spent.

* If they spend over $10,000, you will give them a 40 percent discount.
* Customers will receive a 25 percent discount for purchases over $5000
* Customers will receive a 10 percent discount for purchases over $1000.

The solution below uses if statements to determine the appropriate switch-case code.

Change the value of totalsales in the second line of the program to 99, 150, 1500 and refresh the web page each time – what happens?

<?php

**$totalsales = 15000;**

if ($totalsales<= 0)

{

echo “There were no sales recorded at this time …”;

} else {

if ($totalsales>= 10000) $SwitchCase = 1;

if ($totalsales>= 1000 && $totalsales<=9999) $SwitchCase = 2;

if ($totalsales>= 100 && $totalsales<=999) $SwitchCase = 3;

if ($totalsales<= 100) $SwitchCase = 4;

switch ($SwitchCase)

{

case 1:

echo “ <h2> Thank you for shopping at our online store! </h2>”;

echo “<h3> Please come by again soon… <br>”;

echo “Because your total purchase today was: $totalsales</h3>”;

echo “<h1> We want to give you a 40% discount on your next in-store

purchase with us ! </h1>”;

echo “<h2> Please print this page for your records, and present it at your next visit to our downtown location </h2>”;

break;

case 2:

echo “ <h2> Thank you for shopping at our online store! </h2>”;

echo “<h3> Please come by again soon… <br>”;

echo “Because your total purchase today was: $totalsales</h3>”;

echo “<h1> We want to give you a 25% discount on your next in-store

purchase with us ! </h1>”;

echo “<h2> Please print this page for your records, and present it at your next visit to our downtown location </h2>”;

break;

case 3:

echo “ <h2> Thank you for shopping at our online store! </h2>”;

echo “<h3> Please come by again soon… <br>”;

echo “Because your total purchase today was: $totalsales</h3>”;

echo “<h1> We want to give you a 10% discount on your next in-store

purchase with us ! </h1>”;

echo “<h2> Please print this page for your records, and present it at your next visit to our downtown location </h2>”;

break;

default:

echo “<h2> Thank you for shopping at our online store ! </h2>”;

echo “<h3> Please come by again soon… <br>”;

echo “Your total purchase today was: $totalsales</h3>”;

break;

}

}

?>

**For-next**

Use when you want to repeat a set amount of code a set number of times.

Simple syntax:

For (start condition; end condition; value adjustment)

{

// code to be executed…

}

**Fornextexample.php** To process a tax rate of 10 percent over a range of dollar values from $100 to $5000 in $100 increments, you would use the for-next construct.

<?php

For ( $money = 100; $money <= 5000; $money = $money + 100)

{

// alternately you could write the increment equation as $money += 100

$total = $money + ( $money \* 0.10);

echo "The tax on $money at 10% is $total";

echo "<br>";}

?>

Do you see the following output?

The tax on 100 at 10% is 110  
The tax on 200 at 10% is 220

:

The tax on 4900 at 10% is 5390  
The tax on 5000 at 10% is 5500

Change the tax rate to 12% and the range to 100-5000 in 200 increments.

**Do-While**

Use when you want to repeat a set amount of code while some condition is true.

Simple Syntax

Do {

//code to be executed …

}

while (condition);

**dowhileexample.php**This code runs a counter until the variable $number equals 51. The $number ++’ statement serves as the counter. The execution point then moves to the next line of executable code following the while portion of the loop.

<?php

$number = 1;

do {

echo “ the current number is: $number “;

$number ++;

echo “<br>”;}

while ($number <= 50);

**echo “ the current number is: $number “;**

?>

What do you think happens if you add the line in bold as shown above? Try it.

**Customer’s Status Example.** Using programming flow control, determine a customer’s status with a bank based on their credit rating or existing balance. Use the following table to determine the course of action, and then display the results in a table format. The solution is an html program that obtains a customer’s status code or bank balance and then displays their status level in a form. The form uses the action in a PHP program.

|  |  |  |
| --- | --- | --- |
| Status Code | Balance Amount | Status Level |
| A | >$10,000 | Preferred |
| B | >$1000 | Regular |
| C | >$100 | Caution |
| D | <$100 | Avoid! |

A list box will be used for the entry of a credit code. A single-line entry box will be used to enter the balance. A form will display the results and allow for returning to the main menu.

**The code for enterstatus.html:**

<html>

<head>

<title> Customer Banking Status </title>

</head>

<body>

<h1>Please select a status code or enter a bank balance </h1>

<form method=”post” action=”creditstatus.php”>

<table><tr><td>

Enter the customer’s full name: </td><td></td>

<td><input type=”text” name=”fullname” size=”25”>

</td></tr>

<tr></tr>

<tr><td>

Select a status code</td>

<td></td>

<td>

<select name=”statuscode” size=”1”>

<option value=0> No Status

<option value=1> A

<option value=2> B

<option value=3> C

<option value=4> D

</select></td></tr><tr></tr><tr><td>

or enter a bank balance </td><td>$</td>

<td><input type=”text” name=”balance” size=”12”></td></tr>

<tr><td>(rounded to nearest dollar)</td></tr>

<tr></tr>

</table>

<br><br>

<input type=”submit” value=”Check Status”>

</form>

</body>

</html>

**code for creditstatus.php**

<html>

<head>

<title> Display the customer's bank status </title>

</head>

<body>

<h1>The customer information results in the following: </h1>

<table><tr><td>

<?php

If ($\_POST[statuscode] > 0) {

switch ($\_POST[statuscode]) {

case 1:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Selected status code was: A </td></tr>

<tr><td>

Therefore the credit status is: PREFERRED </td></tr>

<?php break;

case 2:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Selected status code was: B </td></tr>

<tr><td>

Therefore the credit status is: REGULAR </td></tr>

<?php break;

case 3:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Selected status code was: C </td></tr>

<tr><td>

Therefore the credit status is: CAUTION </td></tr>

<?php break;

case 4:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Selected status code was: D </td></tr>

<tr><td>

Therefore the credit status is: AVOID !</td></tr>

<?php break;

}

} elseif ($\_POST[balance] > 0) {

if ($\_POST[balance] >= 10000) $SwitchCase = 1;

if ($\_POST[balance] >= 1000 && $\_POST[balance] <=9999) $SwitchCase = 2;

if ($\_POST[balance] >= 100 && $\_POST[balance] <=999) $SwitchCase = 3;

if ($\_POST[balance] < 100) $SwitchCase = 4;

switch ($SwitchCase){

case 1:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Entered balance was: $<?= $\_POST[balance]; ?></td></tr>

<tr><td>

Therefore the credit status is: PREFERRED </td></tr>

<?php break;

case 2:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Entered balance was: $<?= $\_POST[balance]; ?></td></tr>

<tr><td>

Therefore the credit status is: REGULAR </td></tr>

<?php break;

case 3:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Entered balance was: $<?= $\_POST[balance]; ?></td></tr>

<tr><td>

Therefore the credit status is: CAUTION </td></tr>

<?php break;

case 4:

?>

for customer: <?= $\_POST[fullname]; ?></td>

<td></tr><tr><td>

Entered balance was: $<?= $\_POST[balance]; ?></td></tr>

<tr><td>

Therefore the credit status is: AVOID !</td></tr>

<?php

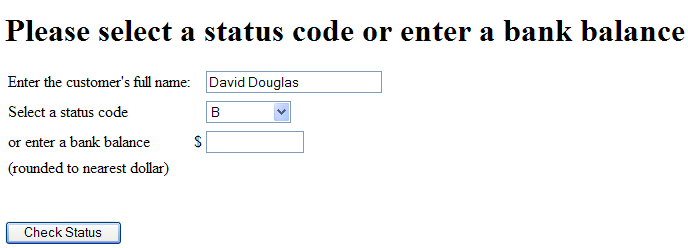
}

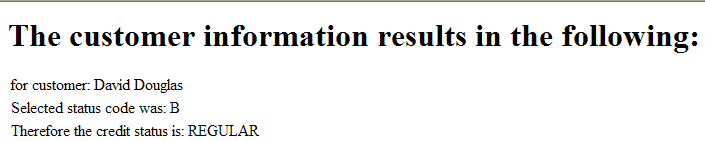
}

?>

</body>

</html>





How was the drop down box created?

Add a Return to Main Menu link on the form.

**Coin Counting Example**. Design a form to calculate the amount of money you have in a change drawer. Allow for the entry of pennies, nickels, dimes, quarters, 50-cent pieces, dollar coins, and two-dollar coins. Show the totals in both cents and dollars in a printable table. Also, provide a way to return to the coin entry page if necessary.

**Code for enterchange.html:**

<html>

<head>

<title> Change Cash Calculator </title>

</head>

<body><h1> Please enter your coin count and denomination </h1>

<br>

<form action="coincount.php" method="post">

Pennies: &nbsp;

<input type=text name="pennies" size=6>&nbsp; &nbsp; &nbsp;

Nickels: &nbsp;

<input type=text name="nickels" size=6><br>

Dimes: &nbsp;

<input type=text name="dimes" size=6>&nbsp; &nbsp; &nbsp;

Quarters: &nbsp;

<input type=text name="quarters" size=6><br>

50 Cent Coins: &nbsp;

<input type=text name="fifty" size=6>&nbsp; &nbsp; &nbsp;

1 Dollar Coins: &nbsp;

<input type=text name="dollar" size=6><br>

2 Dollar Coins: &nbsp;

<input type=text name="twodollar" size=6><br><br>

<Input type=submit name="submit" value="Calculate Coins">&nbsp;&nbsp;

<Input type=reset name="reset" value="Clear form">

</form>

</body>

</html>

**Code for coincount.php**

<html>

<head>

<title> Display the contents of a submitted form </title>

</head>

<body>

<h1>The following coins were counted and valued: </h1>

<?php

$total\_cents = $\_POST[pennies] + ($\_POST[nickels] \* 5) + ($\_POST[dimes] \* 10) + ($\_POST[quarters] \* 25)

+ ($\_POST[fifty] \* 50) + ($\_POST[dollar] \* 100) + ($\_POST[twodollar] \* 200);

$total\_dollars = $total\_cents / 100;

$total\_cents\_fmt = number\_format($total\_cents);

$total\_fmt = number\_format($total\_dollars,2,'.','');

?>

<table><tr><td>Pennies: </td>

<td><?php echo $\_POST[pennies]; ?></td>

<td> Nickels: </td>

<td><?php echo $\_POST[nickels]; ?></td></tr>

<tr><td> Dimes: </td>

<td><?php echo $\_POST[dimes]; ?></td>

<td> Quarters: </td>

<td><?php echo $\_POST[quarters]; ?></td></tr>

<tr><td> 50-cent coins: </td>

<td><?php echo $\_POST[fifty]; ?></td>

<td> 1-dollar coins: </td>

<td><?php echo $\_POST[dollar]; ?></td></tr>

<td> 2-dollar coins: </td>

<td><?php echo $\_POST[twodollar]; ?></td></tr>

</table>

<h3>You therefore have: </h3>

<table><tr><td> Total in Cents: </td>

<td><? echo "$total\_cents\_fmt"; ?></td></tr><tr></tr><tr>

<td> Total Dollars: </td>

<td>$<? echo "$total\_fmt"; ?></td></tr>

</table>

</body>

</html>

