**UFCFR5-15-3: Advanced Topics In Web Development II**

**Workshop (OO Calculator)**

Expected learning outcomes:

* Understand the basic object-oriented programming in PHP
* Develop an object-oriented calculator

**PHP: Processing Form Data, the $\_GET and $\_POST variables and User Defined Functions**

In this tutorial we will use and extend a simple calculator program calculator\_v1.php (see code) which multiplies two numbers and shows the use of a HTML Form and the use of the $\_GET variable.

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 40 41 42 43 44 45 46 47 48 | <?php  /\* ===================================================  PHP Calculator example using "sticky" form (Version 1)  ======================================================  Author : P Chatterjee (adopted from an original example written by C J Wallace)  Purpose: To multiply 2 numbers passed from a HTML form and display the result.   input:   x, y : numbers   calc : submit button  \*/   // grab the form values from $\_POST hash  extract($\_GET);   // first compute the output, but only if data has been input  if(isset($calc)) { // $calc exists as a variable   $prod = $x \* $y;  } else { // set defaults   $x=0;   $y=0;   $prod=0;  }  ?>  <!DOCTYPE html>  <html>  <head>  <title>PHP Calculator (Version 1)</title>  </head>  <body>  <h3>PHP Calculator (Version 1)</h3>  <p>Multiply two numbers and output the result</p>   <form method="get" action="<?php print $\_SERVER['PHP\_SELF']; ?>">  x = <input type="text" name="x" size="5" value="<?php print $x; ?>"/>  y = <input type="text" name="y" size="5" value="<?php print $y; ?>"/>  <input type="submit" name="calc" value="Calculate"/>  </form>   <!-- print the result -->  <?php  if(isset($calc)) {   print "<p>x \* y = $prod</p>";  }  ?>  </body>  </html> |

Note:

1. That the form is a "sticky" form - that is, it handles both the PHP processing and the HTML output; if there are no parameters passed to the script, it simply outputs the default html form. On the other hand, if it detects parameters, it does the calculation and then outputs the form and the result.
2. The use of the PHP built-in **extract()** function on line 14. This function takes as array, in this case $\_GET, and turns any keys into PHP variables and assigns the appropriate values passed with the keys. The form is passing three parameters (examine the URL after pressing the calc button) and the script will have three variables after extract($\_GET)- $x, $y & $calc with their appropraite values assigned.
3. Lines 25-48 is the HTML form.
4. Lines 43-45 checks if $calc is set and prints out the result if it is.

TASKS IN CLASS (1st hour)

1. Run and test the program in your computer

2. Extend the script to perform the four basic arithmetic functions: +, -, \*, / and return a result.

3. Extend the script to move the calculations into a function and make use of this function.

|  |
| --- |
| function calculate($x, $y, $operator) { … } |

**Build an object-oriented calculator**

TASKS IN CLASS (2nd hour)

4. Design a Calculator class using a \_\_construct() method and an appropriate calculate() method.

5. Instantiate a calculator object and test it.

6. Embed the class in your script and use it via the form to get user input.

7. Ensure only numbers are allowed in the x and y fields and you gracefully handle the divide by zero case.