**[IS113] Extra Exercises - Week 4 - Form Processing (with Tutorial Notes)**

**Resource Download**

* To download the **resource files**, go to: <https://bit.ly/2WA9C7l>
* You will see FOUR (4) folders (**hello**, **fruits, kpop, login**).
  + Download & place the folders inside **<web root>/is113/extra4/** on your local computer.
  + You should be able to see the following directory/file structure on your local computer:

o **<web root>/is113/extra4/**:

* hello/ (folder)
  + hello.html (file)
  + sent.php (file)
* fruits/ (folder)
  + images/ (folder)
  + fruit.html (file)
  + sent.php (file)
* kpop/ (folder)
  + images/ (folder)
  + kpop.html (file)
  + sent.php (file)
* login/ (folder)
  + login.html (file)
  + sent.php (file)

|  |  |
| --- | --- |
| **/is113/extra4/hello/hello.html** | **/is113/extra4/hello/sent.php** |
| <html><head><title>Hello Form</title></head>  <body>  <h1>Hello Form</h1>  <hr>  <form action='**sent.php**' method='**GET**'>  Message: <input type='text' name='**msg**'>  <br>  Number: <input type='text' name='**num**'>  <br>  <input type='submit'>  </form>  </body></html> | The user clicks on the **SUBMIT** button in **hello.html** page... and the user is taken to **sent.php**    How do we **retrieve** the **form input parameters** and **values** for **further processing**?    Let’s have a look! ^\_^ |
|  | // When the user clicks on the **SUBMIT** button in **hello.html**  // What is the resulting **URL**?    **http://localhost/is113/extra4/hello/sent.php?msg=Hello&num=3** |
|  | // What **input parameters** and **values** did **hello.html** pass to me?  // Can I print it? Yes, using **var\_dump()**.    var\_dump($\_GET);    // What is the **data type** of $\_GET**?** **Associative Array** |
|  | // What is another way to see what’s inside $\_GET? **print\_r()**  echo '<pre>';  print\_r($\_GET);  echo '</pre>'; |
|  | // How do I print the **value** of the **input parameter msg?**    echo $\_GET['msg']; |
|  | // How do I print the **value** of the **input parameter msg**  // using **Size 1 Heading**?    echo '<h1>' . $\_GET['msg'] . '</h1>';  // Another way (THIS WILL ERROR OUT!!!)  echo "<h1>$\_GET['msg']</h1>"; // ERROR  // Another way (THIS WILL WORK)  // Don’t forget to put **curly braces** around associative array  echo "<h1>{$\_GET['msg']}</h1>"; |
|  | // OH NO!!!  // The user left **msg** and **num** parameter values **EMPTY** !!!  // What’s inside $\_GET associative array?    var\_dump($\_GET);    // OKAY... the two input parameter values look **EMPTY**. |
|  | // If the **msg** input parameter value is **EMPTY**, print...  // "Hey! Got no message for me ah?"  // How do I check if the **input parameter value** is **EMPTY**?  // Use **empty()**    if( empty($\_GET['msg']) == TRUE )  echo "Hey! Got no message for me ah?";    if( empty($\_GET['msg']) )  echo "Hey! Got no message for me ah?";    if( $\_GET['msg'] == '' )  echo "Hey! Got no message for me ah?"; |
|  | // What will the below code print?  if( empty($\_GET['num']) )  echo "Hey! No number ah?";  else  echo "Num: " . $\_GET['num'];    // It will print  **Hey! No number ah?**    // WHAT? I typed number zero (0) into **num** input field !!!  // WHY IS THIS EMPTY?  // Input zero (0) is treated EMPTY  // When check for **NUMBER** input values, be **CAREFUL** !!! |
| In the “Message” input field, I pressed **SPACE BAR** 6 times. As a result, the **msg** input parameter value is NOT an empty string but a **String** of length 6. | // How do check the length of a String? **strlen()**  echo strlen($\_GET['msg']); // prints **6**  // See? It’s not an empty string!  // It contains **SIX (6) whitespace characters**.  // Should we treat such input values as **EMPTY**?  // If YES, then we need a way to **TRIM** such strings. |
|  | // The **trim()** function removes whitespace and other  // predefined characters from both sides of a string.    $str = ' Hello ';  $str2 = trim($str);    echo strlen($str) . '<br>'; // prints **7**  echo strlen($str2); // prints **5** |
|  | // The **rtrim()** function removes whitespace and other  // predefined characters from **right side** of a string.    $str = " Hello ";    $str2 = rtrim($str);    echo strlen($str) . '<br>'; // prints **10**    echo strlen($str2); // prints **6**    // Similarly, **ltrim()** removes from **left side**. |
| In the “Message” input field, I pressed **SPACE BAR** 6 times. As a result, the **msg** input parameter value is NOT an empty string but a **String** of length 6. | // How do I check if the **input parameter value** is **EMPTY**?  // The user typed a string consisting of **6 whitespaces**.    if( trim($\_GET['msg']) == FALSE )  echo "Hey! Got no message for me ah?";    // This also works!  // Boolean **FALSE** is **0**  // Boolean **TRUE** is **1**    if( trim($\_GET['msg']) == 0 )  echo "Hey! Got no message for me ah?"; |
| In the “Message” input field, I pressed **SPACE BAR** 6 times and typed **Hello**. As a result, the **msg** input parameter value is NOT an empty string but a **String** of length 11. | // What will the below code print?    if( trim($\_GET['msg']) == FALSE )  {  echo "Hey! Got no message for me ah?";  }  else  {  $new\_str = **trim**($\_GET['msg']);  echo $new\_str . '<br>';  echo 'Length of this string: ' . strlen($new\_str);  }    // It will print  **Hello**  **Length of this tring: 5**      // Do you see why the length of the **trimmed string** is **5**? |
|  | // How to check if a value is **String**? **is\_string()**  // What will the below code print?  $str = 'Hello';    if( is\_string($str) )  echo "It's a String";  else  echo "It's NOT a string";    // It will print  **It's a String** |
|  | // How to check if a value is **String**? **is\_string()**  // What will the below code print?  $num = 17;    if( is\_string($num) )  echo "It's a String";  else  echo "It's NOT a string";    // It will print  **It's NOT a String** |
|  | // What about **Form Input**? Does **is\_string()** work the same way?  // TEST yourself with $\_GET and note down observations below.  // **HINT**: **var\_dump($\_GET)** and see…  // Is **$\_GET['num']** a **String**? Or, is it **Numeric**? |
|  | // How to check if a value is **Number**? **is\_numeric()**  // What will the below code print?  $num = 17;  if( is\_numeric($num) )  echo "It's a Number";  else  echo "It's NOT a Number";    // It will print  **It's a Number** |
|  | // What about **Form Input**? Does **is\_numeric()** work the same way?  // **HINT**: **var\_dump($\_GET)** and see…  // Is **$\_GET['num']** a **String**? Or, is it **Numeric**?  It is a **String**.  **is\_numeric()** performs implicit type conversion.  **is\_numeric($\_GET['num'])** will return **Boolean TRUE**. |
|  | // How to check if a value is **Integer Number**? **is\_int()**  // What will the below code print?  $num = 17;    if( is\_int($num) )  echo "It's an Integer Number";  else  echo "It's NOT an Integer Number";    // It will print  **It's an Integer Number** |
|  | // How to check if a value is **Integer Number**? **is\_int()**  // What will the below code print?  $num = 17.89;  if( is\_int($num) )  echo "It's an Integer Number";  else  echo "It's NOT an Integer Number";  // It will print  **It's NOT an Integer Number** |
|  | // **IMPORTANT**: **is\_int()** **does NOT** work for **Form Inputs**.  // What is the **variable type of** $\_GET['num']? à **STRING**  // What will the below code print?  if( is\_int($\_GET['num']) )  echo "It's an Integer Number";  else  echo "It's NOT an Integer Number";    // It will print  **It's NOT an Integer Number**  // **HINT**: **var\_dump($\_GET)** and see…  // Is **$\_GET['num']** a **String**? Or, is it **Numeric**?  It is a **String**.  **is\_int()** does NOT perform implicit type conversion!!!  **is\_int($\_GET['num'])** will return **Boolean FALSE** even though the user typed in an integer (it is still treated as a **String**).  ***SO CONFUSING!!! DO NOT USE THIS FUNCTION!!!*** |
|  | // **INSTEAD**: **filter\_var()** works!  // Ref: https://developer.hyvor.com/php/input-validation-with-php  // What will the below code print?    if( **filter\_var**($\_GET['num'], **FILTER\_VALIDATE\_INT**) )  echo "It's an Integer Number";  else  echo "It's NOT an Integer Number";    // It will print  **It's an Integer Number** |
|  | // **INSTEAD**: **ctype\_digit()** works!  // Ref: http://php.net/manual/en/function.ctype-digit.php  // Checks if **all characters** in the provided string are **numerical**.  // What will the below code print?    if( ctype($\_GET['num']) )  echo "It's an Integer Number";  else  echo "It's NOT an Integer Number";    // It will print  **It's an Integer Number** |
| Form Processing  Exercise 1      // See below for test cases. | 1) Assume that the user will NOT mistakenly put whitespaces left/right of the input values. Your PHP code does NOT have to check for whitespaces.  2) Assume that zero (0) is a valid input for **msg**.  3) Assume that zero (0) is a valid input for **num**.  4) Create $errors[] array, which will contain zero or more **error messages**.  a. If the user input nothing (empty value) for **msg**, insert an error message (String) "**Why No Message?**"  b. If the user input nothing (empty value) for **num**, insert an error message (String) "**Why No Number?**"  c. If the value of **num** is **NOT an Integer**, insert an error message (String) **"Num is not an Integer"**.  5) If the $errors[] array contains one or more error messages, display error messages as an **ordered list**.  6) Else (if there are no error messages in the array), display the **msg** value **num** number of times in a **table**. |
|  | // IMPORTANT – Look at the flow of the PHP page  <?php  // 1. Retrieve form input from $\_GET  // 2. Validate (empty? Number?)  // Insert error msgs into **$errors[]** array  // DO NOT PRINT ERROR MSGS HERE USING ECHO!!! WAIT!  ?>  <html><body>  <?php  // 3. Display either TABLE or ERROR msgs  if( count($errors) > 0 ) {  // Got errors  // List error msgs  }  else {  // No errors  // Display msg in <TABLE>  }  ?>  </body></html> |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **/is113/extra4/fruits/fruit.html** | **/is113/extra4/fruits/sent.php** |
|  | // When the user clicks on the **SUBMIT** button in **fruit.html**  // What is the resulting **URL**?    **http://localhost/is113/extra4/fruits/sent.php?fruit=orange&quantity=1** |
|  | // What will the below code print?    var\_dump($\_GET); |
|  | // What will the below code print?    echo "You chose: {$\_GET['fruit']} <br>";  echo "Quantity: {$\_GET['quantity']} ";    // It will print  **You chose: orange**  **Quantity: 1** |
|  | // When the user clicks on the **SUBMIT** button in **fruit.html**  // What is the resulting **URL**?    **http://localhost/is113/extra4/fruits/sent.php?quantity=1**    // **<input type='radio' ... >** works a bit **DIFFERENTLY** from  // **<input type='text' ... >**    If the user **does NOT choose RADIO BUTTON option**,  the **RADIO BUTTON input parameter is NOT PASSED**  to the PHP file at all.    Check URL: Only the input parameter **quantity** is sent to **sent.php** !!! |
|  | // What will the below code print?    var\_dump($\_GET); |
|  | // What will the below code print?  <?php  echo "You chose: {$\_GET['fruit']} <br>";  echo "Quantity: {$\_GET['quantity']} ";  ?>  **// It will ERROR OUT**    $\_GET associative array **does NOT** have an item with key **'fruit' --> PHP ERROR** |
|  | /\* How to **safe-guard** against **invalid user inputs**?  e.g. None of the radio button options are selected!  1) One option is to have one of the radio button options selected by default in **fruit.html**.  2) Suppose that option 1) is not available and you MUST handle (the case of no radio button option selected) on the server side in **sent.php**. \*/  // You can use **isset()** function.  **// It checks whether a variable is set or not !!!** |
|  | // Given an associative array  $fruits = [ 'apple' => 2,  'orange' => 3,  'pear' => 5 ];    // What will the below code print?    if( isset($fruits['apple']) )  echo 'Got apples';  else  echo 'No apples';    // It will print  **Got apples**    // isset() returns **Boolean TRUE** because  // $fruits associative array **HAS** an item with **key** **'apple'** |
|  | // Given the above associative array  // What will the below code print?    if( isset($fruits['mango']) )  echo 'Got mangos';  else  echo 'No mangos';    // It will print  **No mangos** |
|  | // What will the below code print?    if( isset($\_GET['quantity']) )  echo 'Got quantity';  else  echo 'No quantity';    // It will print  **Got quantity** |
|  | // What will the below code print?    if( isset($\_GET['fruit']) )  echo 'Got fruit';  else  echo 'No fruit';    // It will print  **No fruit**    **// This way, you can check to see if the user**  **// selected a radio button option in the HTML form or not.** |
| Form Processing  Exercise 2  // See below for test cases. | 1) If the user fails to select a fruit (apple, orange, pear), display **"Must select fruit"**.  2) Else:  a. Display the selected fruit’s image file **quantity** number of times (next to each other).  b. Image files are located in **/is113/extra4/fruits/images/** |
|  |  |
|  |  |
|  |  |
|  |  |
| Thinking  Point | **http://localhost/is113/extra4/fruits/sent.php?fruit=orange**  **http://localhost/is113/extra4/fruits/sent.php?fruit=orange&abc=1**  **http://localhost/is113/extra4/fruits/sent.php**  // Is it **possible** that a user attempts to access your **sent.php** via  // the above URLs? **YES**  // Consider the following scenario.  **URL: http://localhost/is113/extra4/fruits/sent.php?fruit=orange**  Suppose a user keyed in the above URL.  Below is your **sent.php** code:   |  | | --- | | **sent.php** | | echo $\_GET['fruit']; **// Will this code work?**  echo $\_GET['quantity']; **// Will this code work? Will it break?** |     // How do you think you should handle such cases in your PHP code?  // Will **isset()** help in this case? **YES** |

|  |  |
| --- | --- |
| **/is113/extra4/kpop/kpop.html** | **/is113/extra4/kpop/sent.php** |
|  | // When the user clicks on the **SUBMIT** button in **kpop.html**  // What is the resulting **URL**?    **http://localhost/is113/extra4/kpop/sent.php** |
|  | // When the user clicks on the **SUBMIT** button in **kpop.html**  // What is the resulting **URL**?    **http://localhost/is113/extra4/kpop/sent.php**    // The user selected TWO (2) checkboxes.  // How come the resulting **URL** does not reflect the user input?  // HINT: What is the **FORM METHOD** used in **kpop.html**?  // **GET** versus **POST**  // What are the differences between these two **methods**?  // Take notes below. |
| Form Processing  Exercise 3  // See below for test cases. | 1) If the user does NOT select any K-Pop stars, display:  **"OMG nobody selected"**  2) Else:  a. Display the selected stars’ image files.  b. Image files are located in **/is113/extra4/kpop/images/** |
|  |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **/is113/extra4/login/login.html** | **/is113/extra4/login/sent.php** |
| Form Processing  Exercise 4  // See below for test cases. | 1) **sent.php** code is broken.  2) **Fix it** as per the following test cases. |
| Password: trump123  Confirm Password: trump123 |  |
| Password: trump123  Confirm Password: trump123 |  |
| Password: trump123  Confirm Password: abcdefgh |  |
| Password: <empty>  Confirm Password: <empty> |  |
| Password: trump123  Confirm Password: <empty> |  |
| Password: trump123  Confirm Password: trump123 |  |