

PHP Forms

& Global Variables



Lecture 3

2021-2022

Lecturer: Dezheen Hussein Abdulazeez

E-mail: dezheen.abdulazeez@uod.ac

Computer science Dept_UOD

OUTLINES

- **PHP Forms and User Input**
- **Get vs. Post**
- **Combining HTML and PHP code on a single page**
- **PHP Global Variables**

FORM

- It is used to collect data from the user.
- Example1: The example below displays a simple HTML form with two input fields and a submit button.



A screenshot of a simple HTML form. The form is enclosed in a black rectangular border. It contains three elements: a label 'Name:' followed by a text input field with the placeholder text 'Enter Name'; a label 'Age:' followed by a text input field with the placeholder text 'Enter Age'; and a 'Submit' button at the bottom.

Figure1: A simple HTML FORM

A SIMPLE HTML FORM

Example1: Form Code

```
<html><body>
```

```
<form action="1Welcome.php" method="post">
```

```
  <label for="Name">Name:</label>
```

```
    <p> <input type="text" id="Name" name="name" /> </p>
```

```
  <label for="Age">Age:</label>
```

```
    <p> <input type="text" id="Age" name="age" /></p>
```

```
      <input type="submit" />
```

```
</form>
```

```
</body></html>
```

- In the example HTML page above when the user fills in this form and click on the submit button, the form data is sent to the "**1welcome.php**" file.
- The "**1welcome.php**" file looks like this:

```
<html><body>
```

```
<?php
```

```
    echo "Welcome ". $_POST['name'].".<br >";
```

```
    echo "You are ". $_POST['age'] ." years old.";
```

```
?>
```

```
</body></html>
```

- The output of the above script is:

Welcome Azad.

You are 22 years old.

PHP FORMS AND USER INPUT

- There are two attributes to the <form> tag that you should be aware of and use: action and method.
- **Action** sets the location of the page that will handle the results of the form the place where the variables should be sent.
- **Method** describes how the data should be submitted, and you have two options: GET and POST.
- The PHP \$_GET and \$_POST variables are used to retrieve information from forms, like user input.

PHP \$_GET

- The \$_GET variable is used to collect values from a form with method="get".
- Information sent from a form with the GET method is visible to everyone.
- All variable names and values are displayed in the URL.
- It has limits on the amount of information to send.
- GET should NEVER be used for sending sensitive information like passwords and credit card numbers.

PHP \$_POST

- The \$_POST variable is used to collect values from a form with method="post".
- Information sent from a form with the POST method is invisible to others.
- All names/values are embedded within the body of the HTTP request.
- It has no limits on the amount of information to send.

RETRIEVE MULTIPLE VALUES WITH PHP

- when we working with SELECT elements. These elements make it possible for the user to choose multiple items, like that
 - `<select name="products" multiple>`
- the script that receives this data has access to only a single value corresponding to this name. We can change this behaviour by renaming an element of this kind so that its name ends with an empty set of square brackets as we have in the below example.
 - `<select name="products[]" multiple>`

SELECT ELEMENT

- Example2: Select Element



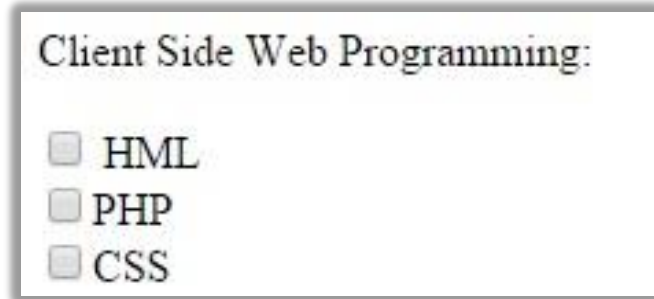
```
<body>  
<label for=" products "> Select some products:</label>  
  <p>  
    <select name= "products[]" multiple>  
      <option value= " Motherboard "> Motherboard</option>  
      <option value= "CD-ROM "> CD-ROM </option>  
      <option value= "Monitor "> Monitor </option>  
      <option value= "Keyboard"> Keyboard </option>  
    </select>  
  </p>  
</body>
```

RETRIEVE SELECT VALUES WITH PHP

```
echo "<p> Your products are: <p>";  
if(!empty($_POST[' products ']))  
{  
    echo"<ul>";  
    foreach($_POST[' products '] as $value){  
        echo"<li>$value</li>";  
    }  
    echo"</ul>";  
}
```

CHECKBOX INPUT TYPE

- Example3: Checkbox Input Type



Client Side Web Programming:

☐ HML

☐ PHP

☐ CSS

<body>

<label for="name">Client Side Web Programming </label>

<p>

<input type="checkbox" name="WebProgram[]" value="html">

HTML

<input type="checkbox" name="WebProgram[]" value="php">

PHP

<input type="checkbox" name="WebProgram[]" value="css">

CSS

</p>

</body>

RETRIEVE CHECKBOX VALUES WITH PHP

```
echo "<p> Client side Markup Language are : <br>";

if(!empty($_POST['WebProgram ']))
{
    echo"<ul>";

    foreach($_POST['WebProgram '] as $value){
        echo"<li>$value</li>";
    }

    echo"</ul>";
}
```

COMBINING HTML AND PHP CODE ON A SINGLE PAGE

Example4: HTML and PHP Code on a same page

```
<html><body>
<form action="<?php echo $_SERVER['PHP_SELF'] ?>" method="post">
    <label for="Name">Name:</label>
    <p> <input type="text" id="name" name="name" /> </p>
    <label for="Age">Age:</label>
    <p> <input type="text" id="Age" name="age" /></p>
    <input type="submit" name="submit" />

</form>
<?php
    if(isset($_POST['submit'])){
        echo "Welcome ". $_POST['name']. "<br >";
        echo "You are ". $_POST['age'] ." years old."; }
?></body></html>
```

OUTPUT



Name:

Age:

Welcome Azad.
You are 22 years old.

Figure2: Combining HTML and PHP Code on a Single Page

- **isset() function** determines if a variable is set and is not NULL.

EXERCISE

E1: Write a PHP code to retrieve the datalist values?



The image shows a web form with a label 'Mobile:' and a dropdown menu. The dropdown menu is open, displaying four options: 'Iphone', 'HTC', 'Nokia', and 'Samsung'. The dropdown menu has a small black triangle pointing downwards on its right side.

Figure3: DataList

PHP GLOBAL VARIABLES - SUPERGLOBALS

- Several predefined variables in PHP are "superglobals", which means that they are always accessible in all scopes and you can access them from any function, class or file without having to do anything special.
- The PHP superglobal variables are:
 1. `$GLOBALS`
 2. `$_SERVER`
 3. `$_REQUEST`
 4. `$_POST`
 5. `$_GET`
 6. `$_FILES`
 7. `$_COOKIE`
 8. `$_SESSION`

PHP \$GLOBALS

- \$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).
- PHP stores all global variables in an array called \$GLOBALS[index]. The index holds the name of the variable.

PHP \$GLOBALS

Example5: Super global variable \$GLOBALS

```
<?php
$a=22;
$b=3;

function add() {
    $GLOBALS['c']= $GLOBALS['a']+ $GLOBALS['b'];
}
add();
echo $c;

?>
```

- c is a variable present within the \$GLOBALS array, it is also accessible from outside the function.

PHP \$_SERVER

- \$_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

Element/Code	Description
<code>\$_SERVER['PHP_SELF']</code>	Returns the filename of the currently executing script
<code>\$_SERVER['SERVER_NAME']</code>	Returns the name of the host server
<code>\$_SERVER['SERVER_PORT']</code>	Returns the port being used on the user's machine to communicate with the web server
<code>\$_SERVER['REQUEST_METHOD']</code>	Returns the request method used to access the page (such as POST Or GET)

PHP \$_SERVER

Example6: Super global variable \$server

```
<?php
```

```
echo $_SERVER['PHP_SELF'];
```

```
echo "<br>";
```

```
echo $_SERVER['SERVER_NAME'];
```

```
echo "<br>";
```

```
echo $_SERVER['SERVER_PORT'];
```

```
echo "<br>";
```

```
echo $_SERVER['REQUEST_METHOD'];
```

```
?>
```

PHP \$_REQUEST

- PHP \$_REQUEST is used to collect data after submitting an HTML form.

PHP \$_REQUEST

Example7: Super global variable \$GLOBALS

```
<html><body>

<form action="<?php echo $_SERVER['PHP_SELF'] ?>" method="post">

    <label for="Name">Name:</label>

    <p> <input type="text" id="Name" name="fname" /> </p>

    <input type="submit" name="submit" />    </form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>
```

PHP \$_POST

- PHP \$_POST is widely used to collect form data after submitting an HTML form with method="post". \$_POST is also widely used to pass variables.

PHP \$_POST

Example8: Super global variable \$GLOBALS

```
<html><body>
```

```
<form action="<?php echo $_SERVER['PHP_SELF'] ?>" method="post">
```

```
    <label for="Name">Name:</label>
```

```
    <p> <input type="text" id="Name" name="fname" /> </p>
```

```
<input type="submit" name="submit" /> </form>
```

```
<?php
```

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
```

```
    // collect value of input field
```

```
    $name = $_POST ['fname'];
```

```
    if (empty($name)) {
```

```
        echo "Name is empty";
```

```
    } else {
```

```
        echo $name;
```

```
    } }
```

```
?>
```

PHP \$_GET

- PHP \$_GET can also be used to collect form data after submitting an HTML form with method="get".
- \$_GET can also collect data sent in the URL.

PHP \$_GET

Example9: Assume we have an HTML page that contains a hyperlink with parameters:

```
<html><body>
```

```
<a href="test_get.php?subject=PHP&web=Web Class">Test $GET</a>
```

```
</body> </html>
```

- When a user clicks on the link "Test \$GET", the parameters "subject" and "web" are sent to "test_get.php", and you can then access their values in "test_get.php" with \$_GET.
- The example below shows the code in "test_get.php":
- ```
<html><body><?php
 echo "Study " . $_GET['subject'] . " at " . $_GET['web'];
?></body></html>
```