PHP Form Validation

Course Code: **CSC 3222** Course Title: WEB TECHNOLOGIES

Dept. of Computer Science
Faculty of Science and Technology

Lab No:	**3**	**Week No:**	**3**	**Semester:**	
Lecturer:	*Supta Richard Philip & richard@aiub.edu*				

Lecture Outline

1. **Learning Objectives**
2. **PHP Form Handling**
    I. **HTTP POST**
    II. **HTTP GET**
    III. **HTTP GET vs HTTP POST**
3. **PHP Form Validation**
4. **Books and References**

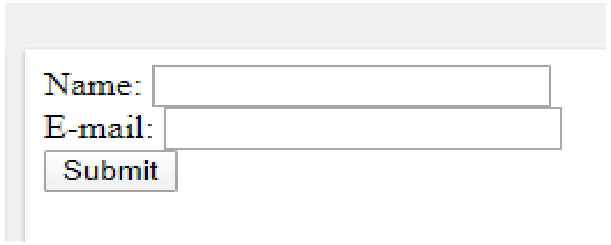
Learning Objectives

- In this Lab, we will learn more details about HTML form elements i.e. different type of form, designing different type of HTML form and form action.
- We will also learn HTTP GET and POST.
- Handling form data using $_GET or $_POST methods and validations form data using PHP.

PHP Form Handling

- The PHP superglobals **$_GET** and **$_POST** are used to collect form-data.
- The example displays a simple HTML form with two input fields and a submit button:

-
  ```
  <html>
  <body>
  <form action="welcome.php" method="post">
  Name: <input type="text" name="name"><br>
  E-mail: <input type="text" name="email"><br>
  <input type="submit">
  </form>
  </body>
  </html>
  ```



PHP Form Handling

HTTP POST

- When the user fills out the form and clicks the submit button, the form data is sent for processing to a PHP file named "welcome.php". The form data is sent with the HTTP POST method.
- welcome.php file

-
  ```
  <html>
  <body>
  Welcome <?php echo $_POST["name"]; ?><br>
  Your email address is: <?php echo $_POST["email"]; ?>
  </body>
  </html>
  ```

PHP Form Handling

HTTP GET

- The same result could also be achieved using the HTTP GET method:
- Welcome_get.php file

-
  ```
  <html>
  <body>
  Welcome <?php echo $_GET["name"]; ?><br>
  Your email address is: <?php echo $_GET["email"]; ?>
  </body>
  </html>
  ```

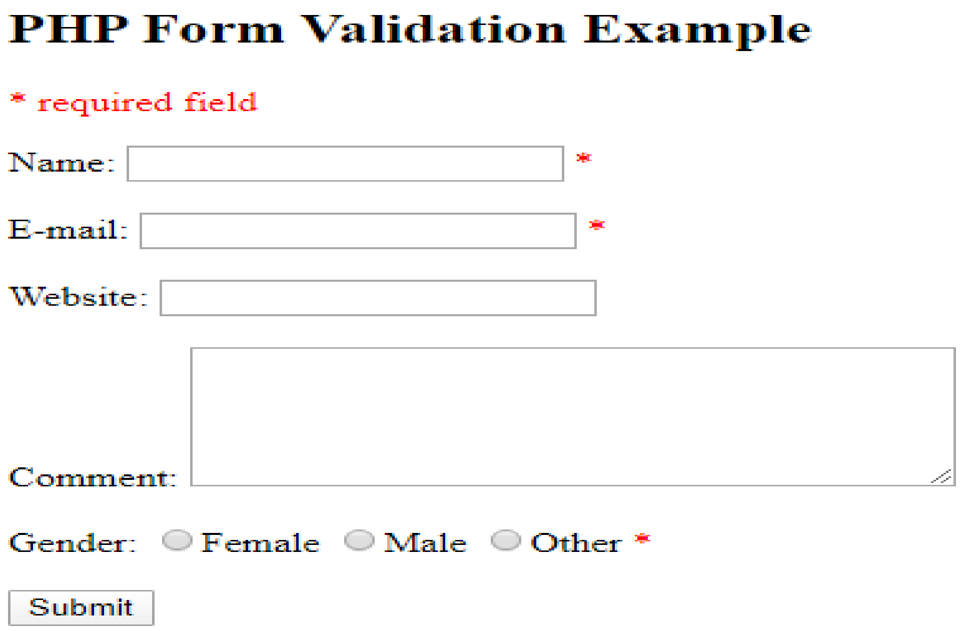
PHP Form Handling

HTTP GET vs HTTP POST

- Both GET and POST create an array (e.g. array(key1 => value1, key2 => value2, key3 => value3, ...)). This array holds key/value pairs, where keys are the names of the form controls and values are the input data from the user.
- $_GET is an array of variables passed to the current script via the URL parameters.
- Information sent from a form with the GET method is **visible to everyone** (all variable names and values are displayed in the URL).
- $_POST is an array of variables passed to the current script via the HTTP POST method.
- Information sent from a form with the POST method is invisible to others

PHP Form Validation

- Proper validation of form data is important to protect your form from hackers and spammers!
- Lets consider the example.



PHP Form Validation Example

* required field

Name: *

E-mail: *

Website:

Comment:

Gender: ○ Female ○ Male ○ Other *

Submit

PHP Form Validation

- The validation rules for the form above are as follows:
- **$_SERVER["PHP_SELF"]** is a super global variable that returns the filename of the currently executing script and sends the submitted form data to the page itself, instead of jumping to a different page.
- `<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">`

Field	**Validation Rules**
Name	Required. + Must only contain letters and whitespace
E-mail	Required. + Must contain a valid email address (with @ and .)
Website	Optional. If present, it must contain a valid URL
Comment	Optional. Multi-line input field (textarea)
Gender	Required. Must select one

PHP Form Validation

- **$_SERVER["PHP_SELF"]** is a super global variable that returns the filename of the currently executing script and sends the submitted form data to the page itself, instead of jumping to a different page.
- `<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">`
- The htmlspecialchars() function converts special characters to HTML entities. This prevents attackers from exploiting the code by injecting HTML or Javascript code (Cross-site Scripting attacks) in forms.
- The validation rules for the form above are as follows:

Field	**Validation Rules**
Name	Required. + Must only contain letters and whitespace
E-mail	Required. + Must contain a valid email address (with @ and .)
Website	Optional. If present, it must contain a valid URL
Comment	Optional. Multi-line input field (textarea)
Gender	Required. Must select one

PHP Form Validation

- Validate the form and display the error message. We will see the details in Lab session.
-

```
<?php
$nameErr = "";
$name =  "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
  }
}


function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
  ?>
```

Books

- W3Schools Online Web Tutorials; URL: http://www.w3schools.com
- PHP Documentation; URL: http://www.php.net/docs.php
- Sams Teach Yourself Ajax JavaScript and PHP All in One; Phil Ballard and Michael Moncur; Sams Publishing; 2010
- JavaScript Phrasebook; Christian Wenz; Sams Publishing; 2007
- PHP and MySQL Web Development, 4/E; Luke Welling and Laura Thomson; Addison-Wesley Professional; 2009
- JavaScript for Programmers Paul J. Deitel and Harvey M. Deitel; Prentice Hall; 2009
- Beginning PHP5, Apache, and MySQL Web Development; Elizabeth Naramore, Jason Gerner, Yann Le Scouarnec, Jeremy Stolz and Michael K. Glass; Wiley Publishing; 2005
- XML in a Nutshell, 3/E; Elliotte Rusty Harold and W. Scott Means; O'Reilly Media; 2004

References

1. https://www.w3schools.com/php/php_forms.asp