

Lecture 5.1

PHP Handling Forms, PHP Super Global variables

Course Instructor
Engr. Madeha Mushtaq

Server side Scripting Languages

- Server side scripts run on a web server.
- Server side scripts are used to:
 - Customize a web page and dynamically change its contents.
 - Respond to queries from users/HTML Forms.
 - Access database, retrieve contents from the database and send information back to browser.

Server side Scripting Languages

- Examples of server side scripting languages:
- PHP
- ASP.Net
- Python
- Perl

PHP

- PHP is a loosely typed language.
- Interpreted language, scripts are parsed at run-time rather than compiled beforehand.
- Open-source – Anyone may view, modify and redistribute source code.
- Platform independent, you can deploy your system on Windows, Linux, Mac OS etc.

PHP

- Through PHP we can develop dynamic websites.
- **Static Websites:** Those websites which structure contents are not changing. User's can't bring changing to the contents.
- **Dynamic Websites:** those websites, whose structure and contents are changing e.g. facebook.com (user upload pictures, links, videos, comments).
- User is interacting with the website and change the contents or modify the contents.

Handling Forms

- Forms provide a mean of submitting information from the client to the server.
- We can create HTML forms using `<form>` tag
- Method and action are the most common attributes of `<form>`.

Handling Forms

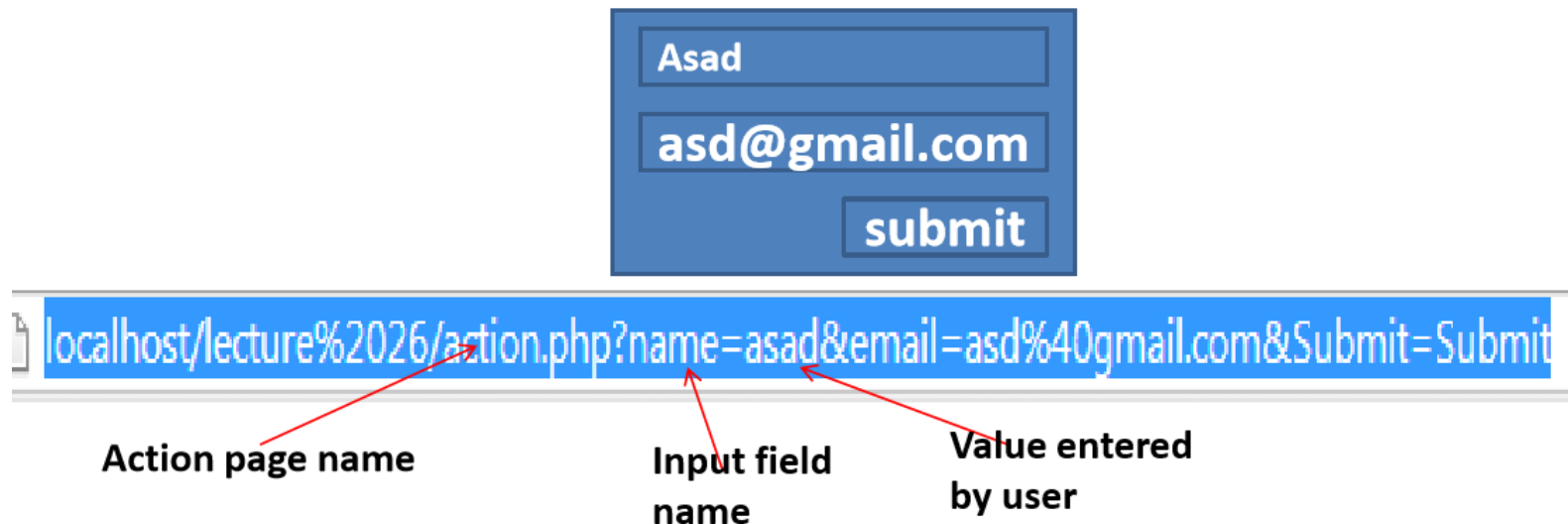
- **action** - gives the URL of the application that is to receive and process the form's data.
- **method** - sets the HTTP method that the browser uses to send the form's data to the server for processing, tells the browser how to send the form data.
 - most common methods are **POST or GET**

Passing form data using GET Method

- Get is useful for sending small amounts of data, (There is an upper limit).
- Get method is not recommended for sending sensitive information like passwords etc.

Passing form data using GET Method

- Get method : All form data is encoded into the URL



Passing form data using Post Method

- Post method can send much larger amounts of form data.
- Data sent by POST method goes through HTTP header, so by using secured HTTP, we can make sure our data is secured.

Passing form data using Post Method

- Post method: form data appears within the message body of the HTTP request.



Welcome Mr. Asad Mahmood

Super Global Variables

- PHP automatically makes few variables available in your program.
- These variables are called super-global variables because they can be accessed without regard to scope.
- A super global variable is a built - in PHP variable that is available in any scope: at the top level of your script, within a function, or within a class method.

Super Global Variables

- `$GLOBALS`
- `$_SERVER`
- `$_REQUEST`
- `$_GET`
- `$_POST`
- `$_FILES`
- `$_COOKIE`
- `$_SESSION`

Super Global Variables

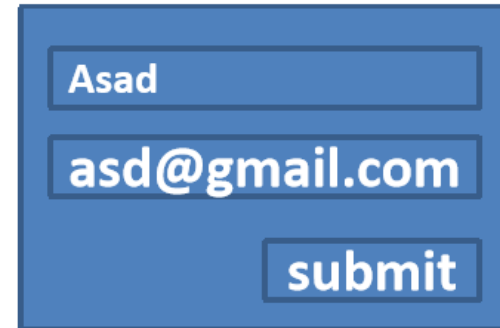
- **\$_GET:** Contains a list of all the field names and values sent by a form using the get method
- **\$_POST:** Contains a list of all the field names and values sent by a form using the post method.
- **\$_REQUEST:** Contains the values of both the \$_GET and \$_POST arrays combined, along with the values of the \$_COOKIE super global array.

Super Global Variables

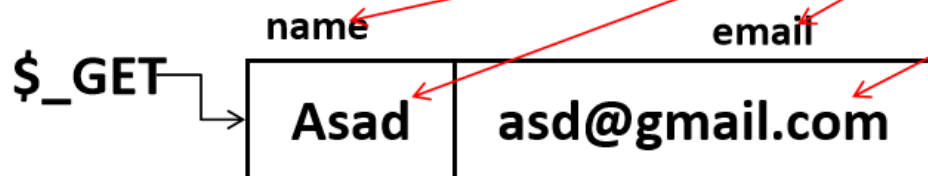
- For example, look at the following form element:
- `< input type="text "name="emailAddress"value="" />`
- You could then access the value that the user entered into that form field using either the `$_GET` or the `$_REQUEST` superglobal:
 - `$email = $_GET["emailAddress"];`
 - `$email = $_REQUEST["emailAddress"];`

Super Global Variables

```
<body>
<form method="get"
action="action.php">
<input type="text" name="name">
<input type="text" name="email">
<input type="submit">
</form>
</body>
```

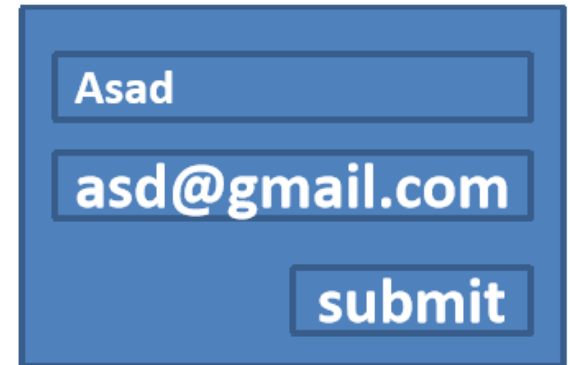


localhost/lecture%2026/action.php?name=asad&email=asd%40gmail.com&Submit=Submit



Super Global Variables

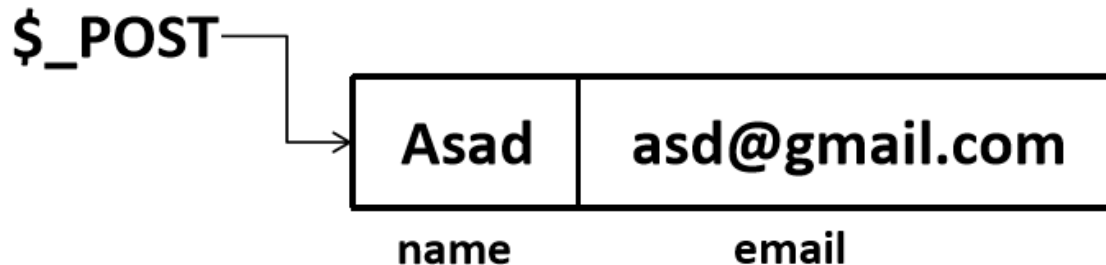
```
<body>  
<form method="post">  
<input type="text" name="name">  
<input type="text" name="email">  
<input type="submit">  
</form>  
</body>
```



Asad

asd@gmail.com

submit



Accessing form data on **action page**

`$_GET` →

name	email
Asad	asd@gmail.com

Asad
asd@gmail.com
submit

Ation.php

```
<?php
```

```
$name = $_GET['name'];
```

```
$email = $_GET['email'];
```

```
?>
```

Accessing form data on action page

\$_POST →

name	email
Asad	asd@gmail.com

Ation.php

```
<?php
```

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

```
$email = $_POST['email'];
```

```
?>
```

Passing text field data

```
<html>
<head>
  <title>Get Method </title>
</head>
<body>
  <form name = "form1" method = "get" action = "action.php">
    Enter your name: <input type = "text" name = "name" value = "Your name" /> <br>
    <input type = "submit" />
  </form>
</body>
</html>
```

Passing text field data

- action.php

```
<?php
$name = $_GET['name'];
echo "Welcome Miss. $name";
?>
```

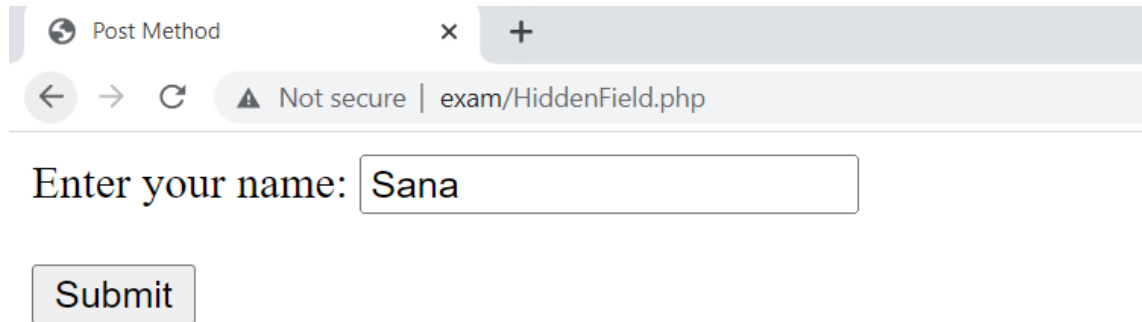
Passing hidden field data

```
<html>
<head>
  <title>POST Method </title>
</head>
<body>
  <form name = "form1" method = "post" action = "action_hidden.php">
    Enter your name: <input type = "text" name = "name" value = "Your name" /> <br>
    <input type = "hidden" name = "hname" value = "Sara" /> <br>
    <input type = "submit" />
  </form>
</body>
</html>
```

action_hidden.php:

```
<?php
$name = $_POST['name'];
echo "Welcome Miss. $name";
echo "<br>";
echo "Hidden name is ". $_POST['hname'];
?>
```

Passing hidden field data

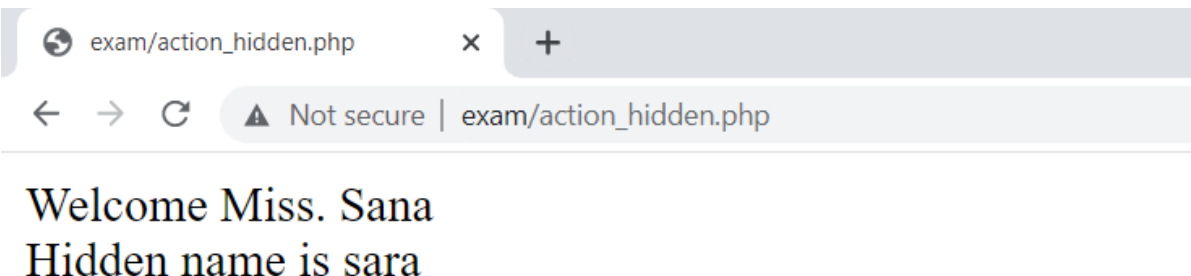


A screenshot of a web browser window. The address bar shows a POST request to `exam/HiddenField.php`. The page content includes a text input field with the value "Sana" and a "Submit" button.

Post Method x +

← → ↻ ⚠ Not secure | exam/HiddenField.php

Enter your name:



A screenshot of a web browser window showing the response to the form submission. The address bar shows the request to `exam/action_hidden.php`. The page content displays a welcome message and a hidden field value.

exam/action_hidden.php x +

← → ↻ ⚠ Not secure | exam/action_hidden.php

Welcome Miss. Sana
Hidden name is sara

Getting value from radio button

```
1 <HTML>
2 <HEAD>
3   <TITLE>Post Method </TITLE>
4 </HEAD>
5 <BODY>
6 <FORM name = "form1" method = "post" action = "action_RadioButton.php">
7   Select your gender? <br>
8   <input type="radio" name="gender" value="Male">Male
9   <input type="radio" name="gender" value="Female">Female
10  <INPUT TYPE="Submit">
11 </FORM>
12 </BODY>
13 </HTML>
```

action_RadioButton.php

```
1 <?php
2   if(isset($_POST['gender']))
3   {
4     echo "You have selected :".$_POST['gender'];
5   }
6 >
```


Dealing with Multi-Value Fields

- We can create form fields that send multiple values.
- To handle multi - value fields in PHP scripts add square brackets (`[]`) after the field name in your HTML form.
- PHP creates a nested array of values within the `$_GET` or `$_POST` (and `$_REQUEST`) super global array.

Getting value from checkbox

```
1 <HTML>
2 <HEAD>
3 <TITLE>Post Method </TITLE>
4 </HEAD>
5 <BODY>
6 <FORM name = "form1" method = "post" action = "action_checkbox.php">
7   Which of the following languages do you know? <br>
8   <INPUT TYPE="checkbox" name = "checklist[]" value = "C" /> C/C++<br>
9   <INPUT TYPE="checkbox" name = "checklist[]" value = "VB" /> VB<br>
10  <INPUT TYPE="checkbox" name = "checklist[]" value = "Java" /> Java<br>
11  <INPUT TYPE="Submit">
12 </FORM>
13 </BODY>
14 </HTML>
```

action_checkbox.php

```
1 <?php
2 if(!empty($_POST['checklist'])){
3   // Loop to store and display values of individual checked checkbox.
4   foreach($_POST['checklist'] as $selected){
5     echo $selected."<br>";
6   }
7 }
8 ?>
```

Getting value from checkbox

Post Method x +

← → ↻ ⚠ Not secure | exam/checkbox.php

Which of the following languages do you know?

☒ C/C++

☐ VB

☒ Java

Submit

exam/action_checkbox.php x +

← → ↻ ⚠ Not secure | exam/action_checkbox.php

C

Java

Getting value from select list

```
<BODY>
<FORM name = "form1" method = "post" action = "action_selectlist.php">
  <select name="Color[]" multiple> // Initializing Name With An Array
    <option value="Red">Red</option>
    <option value="Green">Green</option>
    <option value="Blue">Blue</option>
    <option value="Pink">Pink</option>
    <option value="Yellow">Yellow</option>
  </select>
  <INPUT TYPE="Submit">
</FORM>
</BODY>
</HTML>
```

action_selectlist.php

```
<?php
foreach ($_POST['Color'] as $select)
{
  echo $select. "<br>"; // Displaying Selected Value
}
?>
```

Getting value from select list

Name of the list

Your Domicile:

<select name="dom">

Option and value

<option value="punjab">Punjab</option>

<option value="KPK">KPK</option>

<option value="sindh">Sindh</option>

<option value="Balochistan">Balochistan</option>

</select>

```
echo "<br>";  
echo $_POST[ 'dom' ] ;  
?>
```

Handling Empty Form Fields

- When nothing is sent at all for a field, PHP doesn't create an element for the field in the `$_POST` , `$_GET` , or `$_REQUEST` array.
- If you attempt to access the element, you 'll generate a PHP notice along the lines of:
 - PHP Notice: Undefined index: gender in `action_RadioButton.php` on line 4.
- **Use PHP functions such as `isset()` or `array_key_exists()`.**

References

- Chapter 9, “Beginning PHP6,Apache,Mysql web development” by Matt Doyle, Wrox publishers, 2009, ISBN: 0470413964
- Chapter 13 “Beginning PHP and MySQL” by W. Jason Gilmore, Apress publisher, 4th edition; 2010, ISBN-13 (electronic): 978-1-4302-31158.