Swami Sahajanand College of Computer Science

B.C.A. SEM - V

Subject: Web Application Development Using PHP

UNIT 3

Working with form, Cookie and Session

- Form elements- TextBox, TextArea, Password, RadioButton, Check Box, Combo Box, Image
- Buttons – Submit and Reset
- Uploading File to webserver
- POST & GET method
- PHP include and require statement
- Basic of Cookie-Setting Cookies, Accessing Cookies, Deleting Cookies.
- Basic of Session- Starting a Session, Destroying a session

Write about HTML Forms.

- HTML forms give you the opportunity to gather information from people reading your Web page.
- HTML forms are used to pass inputted data to a server.
- An HTML form can contain input elements like text fields, checkboxes, radio-buttons, submit buttons and more.
- A form can also contain select lists, textarea, fieldset, legend, and label elements.
- To create a form in html page <form> tag is used.
- A HTML Form may contain on or more above listed elements.
- HTML form can be as simple or complex as per requirement.
- One create forms by providing a number of fields in which a user can enter information or choose an option.
- When the user submits the form, the information is returned to a server-side script.
- **A script is a short program that is written specifically for each form**.
- ACTION This attribute points the form to an URL that will accept the form's information and do something with it. If you don't specify an ACTION, it sends the information back to the same URL the page came from.
- METHOD This attribute tells the form how to send its information back to the script. The most common method is POST,

1) Input Tag

- The <input> tag specifies an input field where the user can enter data.
- <input> elements are used within a <form> element to declare input controls that allow users to input data.
- An input field can vary in many ways, depending on the type attribute.
- Input tag is used to add TextBox, CheckBox, RadioButton, hidden Field, File Upload Control etc in form tag.
- The attributes for the <INPUT> tag are the following:
- NAME
  - This defines the name for the data. This field is required for all the types of input except Submit and Clear.
- SIZE
  - This is the size of the input field in number of characters for text or password.
- MAXLENGTH
  - This specifies the maximum number of characters to be allowed for a text or password field.
- VALUE
  - For a text or password field, it defines the default text displayed. For a check box or radio button, it specifies the value that is returned to the server if the box or button is selected. For the Submit and Reset buttons, it defines the text inside the button.
- CHECKED
  - This sets a check box or radio button to on. It has no meaning for any other type of
- <INPUT> tag. TYPE This sets the type of input field you want to display. (See the types in the following section.)
- The most common input types are described below.

2) Text Fields

<input type="text"> is used to accept one line of text from user.

```
<form>
First name: <input type="text" name="firstname"><br>
Last name: <input type="text" name="lastname">
</form>
```

- How the HTML code above looks in a browser:
- Top of Form
- First name: [ ]
  Last name: [ ]
- Bottom of Form
- **Note:** The form itself is not visible. Also note that the default width of a text field is 20 characters.

3) Password Field

- <input type="password"> is used to accept password from user. The characters in a password field are masked (shown as asterisks or circles). It means the input given by user in password field is not readable to others.

```
<form>
Password: <input type="password" name="pwd">
</form>
```

- How the HTML code above looks in a browser:
- Top of Form
- Password: [**********]

4) Radio Buttons

- <input type="radio"> defines a radio button. Radio Button are used to provide a list of choice to user from which user can select only one choice at a time. Radio button can be only selected mutually only if the names of radio buttons are same. However the value of radio button must be different from each other.

```
<form>
<input type="radio" name="rdogender" value="male">Male<br>
<input type="radio" name="rdogender" value="female">Female
</form>
```

- How the HTML code above looks in a browser:
- Top of Form
- ○ Male
  ○ Female

5) Checkboxes

- <input type="checkbox"> defines a checkbox. Checkbox are used to provide a list of choice to user from which user can select multiple choice at a time. Each checkbox must have value attribute in order to pass the value of checked checkbox to server.

```
<form name="Manish_Dholakiya">
<input type="checkbox" name="vehicle" value="Bike">I have a bike<br>
<input type="checkbox" name="vehicle" value="Car">I have a car
</form>
```

- How the HTML code above looks in a browser:
- Top of Form
- ☑ I have a bike
  ☐ I have a car
- Bottom of Form

6) Submit Button

- <input type="submit"> defines a submit button.
- A submit button is used to send form data to a server. The data is sent to the page specified in the form's action attribute.
- The file defined in the action attribute usually does something with the received input:
- <form name="input" action="manish_dholakiya.php" method="get">
  Username: <input type="text" name="user /">
  <input type="submit" value="Submit" />
  </form>

7) Reset Button

- The Reset object represents a reset button in an HTML form.
- Reset Button clear all the input controls to their default values.
- For each <input type="reset"> tag in an HTML form, a Reset object is created.

```
<input type="submit" value="clear all value" />
```

8) image

- Define an image as a submit button:
- The input element with a type attribute whose value is "image" represents either an image from which enables a user to interactively select a pair of coordinates and submit the form,
- The coordinates of the selected point are measured in pixel units from the upper-left corner of the image, and are returned (along with the other contents of the form) in two name/value pairs.
- The x-coordinate is submitted under the name of the field with *.x* appended, and the y-coordinate is submitted under the name of the field with *.y* appended.
- Example →

```
<input type="image" src="img_of_manishdholakiya.gif" alt="Photo of Manish Dholakiya will Display Here">
```

9) TEXTAREA

- <TEXTAREA>, is used to provide a field for someone to enter multiple lines of information.
- By default, a <TEXTAREA> form shows a blank field four rows long and 40 characters wide.
- The default size of TEXTAREA can be change using ROWS and COLS attributes.

- One can also specify some default text by simply entering it between the <TEXTAREA> and </TEXTAREA> tags.
- The options for the <TEXTAREA> tag are as follows:
- NAME This is required. It defines the name for the data.
- ROWS This sets the number of rows in the field.
- COLS This sets the width of the field in characters.

```
<textarea rows="10" cols="30">
We are learning HTML by ManishSir
</textarea>
```

10) SELECT

- The <SELECT> element shows a list of choices in either a pop-up menu or a scrolling list.
- It's set up as an opening and closing tag with a number of choices listed in between.
- Just like the <TEXTAREA> element, the <SELECT> tag requires you to define a name.
- You can specify how many choices to show at once by using the SIZE attribute.
- The options for the <SELECT> element are as follows:
- NAME This is required. It defines the name for the data.
- SIZE This attribute determines how many choices to show. If you omit SIZE or set it to 1, the
- choices are shown as a drop-down list. If you set it to 2 or higher, it shows the choices in a
- scroll box. If you set SIZE larger than the number of choices you have within <SELECT>, a nothing choice is added. When the end user chooses this, it's returned as an empty field.
- MULTIPLE This allows multiple selections. If you specify multiple, a scrolling window displays regardless of the number of choices or the setting of SIZE.

ComboBox

```
<select name="cars">
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
```

ListBox

```
<select name="cars" size='3' multiple='multiple'>
<option value="volvo">Volvo</option>
<option value="saab">Saab</option>
<option value="fiat">Fiat</option>
<option value="audi">Audi</option>
</select>
```

Q-2 How to upload file on server? (Uploading file to server)

- Many time user needs to accept & upload raw input from user. Raw inputs means any type of file like word, excel, power point, zip, audio, video or image type or any other type of file.
- To accept raw input (files) from user one need to use input tag available in HTML.
- To accept raw input (file) one must give type="file" in input tag along with name attributes.
- One also need to set method="post" in form tag.
- User can give both raw input & textual input in single form tag of HTML page.
- After giving all input in form tag, user click on submit button in form tag and then all input will be submitted to webpage whose URL is given action attribute of form tag.
- File accepted from user will be uploaded on remote/local web server's temp directory with temporary name assigned to it by web server.
- Developer has to write a PHP code to move uploaded file from temp directory to his/her project directory.
- Below is sample code to accept file type input from user.

```
<html>
	<head>
		<title> File Upload example</title>
	</head>
	<body> Upload File of Manish Dholakiya: <br>
		<form  action="upload.php"  method="post"  enctype="multipart/form-data">
			Select File <input type="file" name="file1" /><br/>
		<input type="submit" value="Upload" />
		</form>
	</body>
<html>
```

Q-3 Explain POST & GET method.

- The post and get method are two value which used to specify the data transfer method of data submitted by form tag.
- It is used in following way

```
<form method="post" action="mypage.php">
Enter Your name <input type="text" name="txtname" />
</form>
```

- Or

```
<form method="get" action="mypage.php">
Enter Your name <input type="text" name="txtname" />
</form>
```

- Now let us see them in each detail.
- The method attribute specifies how to send form-data (the form-data is sent to the page specified in the action attribute).
- The form-data can be sent as URL variables (with method="get") or as HTTP post transaction (with method="post").

1) GET

- HTTP GET request method is designed to retrieve information from the server. As part of a GET
- Appends form-data into the URL in name/value pairs
- The length of a URL is limited (about 3000 characters)
- Never use GET to send sensitive data! (will be visible in the URL)
- Useful for form submissions where a user want to bookmark the result
- As the data transfers through address bar (URL) there are some restrictions in using space, some characters like ampersand (&) etc in the GET method of posting data. We have to take special care for encoding (while sending) and decoding (while receiving) data if such special characters are present.
- Login details with password should never be posted by using GET method.
- GET is better for non-secure data, like query strings in Google

2) POST

- It is often used when uploading a file or submitting a completed web form.
- Appends form-data inside the body of the HTTP request (data is not shown is in URL)
- Has no size limitations and specially used when submitting raw input (files) to server.
- Form submissions with POST cannot be bookmarked

Q-4 Explain the use of include and require statements in detail.

- **Introduction**
- The 'include (or require)' statement takes all the text/code/markup that exists in the specified file and copies it into the file that uses the include statement.
- Including files is very useful when you want to include the same PHP, HTML, or text on multiple pages of a website.
- It is possible to insert the content of one PHP file into another PHP file (before the server executes it), with the include or require statement.
- The include and require statements are identical, except upon failure:
  - require will produce a fatal error (E_COMPILE_ERROR) and stop the script
  - include will only produce a warning (E_WARNING) and the script will continue
- So, if you want the execution to go on and show users the output, even if the include file is missing, use the include statement.
- Otherwise, in case of FrameWork, CMS, or a complex PHP application coding, always use the require statement to include a key file to the flow of execution.
- This will help avoid compromising your application's security and integrity, just in-case one key file is accidentally missing.
- Including files saves a lot of work. This means that you can create a standard header, footer, or menu file for all your web pages.
- Then, when the header needs to be updated, you can only update the header include file.
- The require statement is also used to include a file into the PHP code.
- However, there is one big difference between include and require; when a file is included with the include statement and PHP cannot find it, the script will continue to execute.
-
- In the following example we use the require statement, the rest statements including echo statement "This will not show" will not be executed because the script execution dies after the require statement returned a fatal error.
- Syntax

include 'filename'; or require 'filename';

- Example

```
<html>
<body>
<?php include 'header.php';?>
<h1>Welcome to my home page!</h1>
<?php require 'noFileAvailable.php';
echo “This will not show!”;
?>
<?php include 'footer.php';
        echo “Design By Manish Dholakiya”; ?>
</body></html>
```

Q-5 What is Cookies? Explain in detail.

Introduction

- All browser and web server use request/response paradigm.
- This means browser generate request, this is forwarded to web server via the internet.
- The web server immediately responds to this request and then forgets about the browser and its request.
- This is how request response paradigm is operated.
- This is not suitable for the website who wishes to conduct any type of business online.
- Browser on other hand require to use this unique identification value when it makes request for the resources on the same server.
- This is where we require to use concept of cookies.
- ***“Cookies are small text file that are created by PHP program run at the server.”***
- Cookies take place in client computer’s memory.
- A cookie is a message given to a Web browser by a Web server.
- The browser stores the message in a text file. The message is then sent back to the server each time the browser requests a page from the server.

How Cookies can be use?

- The main purpose of cookies is to identify users and possibly prepare customized Web pages for them.
- When you enter a Web site using cookies, you may be asked to fill out a form providing such information as your name and interests. This information is packaged into a cookie and sent to your Web browser which stores it for later use.
- The next time you go to the same Web site, your browser will send the cookie to the Web server.
- The server can use this information to present you with custom Web pages.
- So, for example, instead of seeing just a generic welcome page you might see a welcome page with your name on it.

Where Cookies can be use?

1. Shopping cart application
2. Online banking
3. Generation of visitor’s profile
4. Fee based services
5. Website tracking
6. To set remember username and password on Login Page.

Cookies have following information?

- A name of the cookies.
- The IP of the web server from where the cookie originated.
- A unique ID number.
- A date and time stamp.
- Any other relevant text based information.
- This all information is saved as text file on client machine's special sub directory, because of cookie web server will be able to know that new request have been sent by which browser.

How to create cookie?

- PHP cookies are created using the setcookie() function.
- All cookie data is stored in the PHP $_COOKIE global variable and accessible to subsequent pages.

Syntax

setcookie(name, value, expire, path, domain);

Example

```
<?php
setcookie("user", "Manish_Dholakiya", time()+3600);
?>
```

How to retrive cookie?

- PHP cookies are retrieved using the $_COOKIE[] function.

Syntax

$_COOKIE[`<cookie name>`];

Example

```
<?php
        // Print a cookie
echo $_COOKIE["user"];

// A way to view all cookies
print_r($_COOKIE);
?>
```

Q-6 Explain Session in detail.

Introduction

- Session is a way of preserving the user information over different pages.
- In a Dynamic website where user interaction are there and some real application where user manages his/her records that time we need to store some temporary information regarding user identification for that session variable is used.
- *A session variable is a special type of variable whose value is maintained across subsequent web pages.*
- Session variables normally exist until one of the follow criteria is met:
- 1. The user closes the browser window;
- 2. The maximum time allotment set on the server for session lifetime is exceeded;
- 3. Using PHP functions such as session_destroy() to free all session variables currently registered.
- Here there are some simple steps for session
  - Start Session
  - Use/Set the Session
  - Close(Destroy) the Session

1) Start Session

- To use the session information session must be started.
- To start the session **session_start()** function is used.

Syntax/Ex

```
session_start();
```

2) Use/Set Session

- We can use or set the session value using the **$_SESSION[]** function.

Syntax

For Set the value

```
$_SESSION[‘<variable name>’]=<value to be set>
```

For Access the value

```
$<varname>=$_SESSION[‘<variable name>’]
```

Example

For Set the value

```
$_SESSION[‘id’]=101
```

For Access the value

```
$Id=$_SESSION[‘id’];
Echo “Id is :”.$Id;
```

3) Close The Session

- To close the session **session_destroy()** function is used.

Syntax/Ex

```
session_destroy();
```

4) Session_id()

- In PHP, the unique session id value can be retrieved using the session_id() function.
- A unique session_id() value is maintained for each user and is stored in the PHP/session data sub directory located on the Web server.
- Since the session_id() value is unique for all users, it can be used to identify users without the need to create individual user names and passwords. The session_id() function is defined below:
- session_id() - used to get the id value for the current session.

Syntax/Ex

```
session_id();
```