

Lecture 5.2

PHP Super Global variables

\$_COOKIE, \$_Session, \$_File

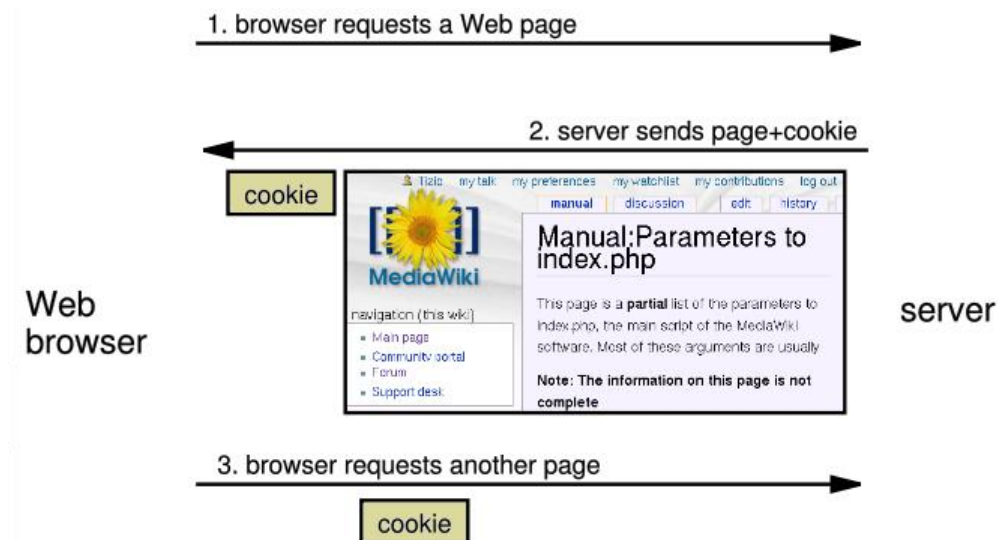
Course Instructor
Engr. Madeha Mushtaq

Cookie

- Cookie is a small amount of information sent by a server to a browser, and then sent back by the browser on future page requests.
- Cookies have many uses:
 - authentication
 - user tracking
 - maintaining user preferences, shopping carts, etc.
- A cookie's data consists of a single name/value pair, sent in the header of the client's HTTP GET or POST request.

How cookies are sent?

- when the browser requests a page, the server may send back a cookie(s) with it.
- If your server has previously sent any cookies to the browser, the browser will send them back on subsequent requests.



How long does a Cookie exist?

- **session cookie** : the default type; a temporary cookie that is stored only in the browser's memory.
 - when the browser is closed, temporary cookies will be erased
 - can not be used for tracking long-term information
 - safer, because no programs other than the browser can access them.
- **persistent cookie** : one that is stored in a file on the browser's computer.
 - can track long-term information
 - potentially less secure, because users (or programs they run) can open cookie files, see/change the cookie values, etc.

Myths about cookies

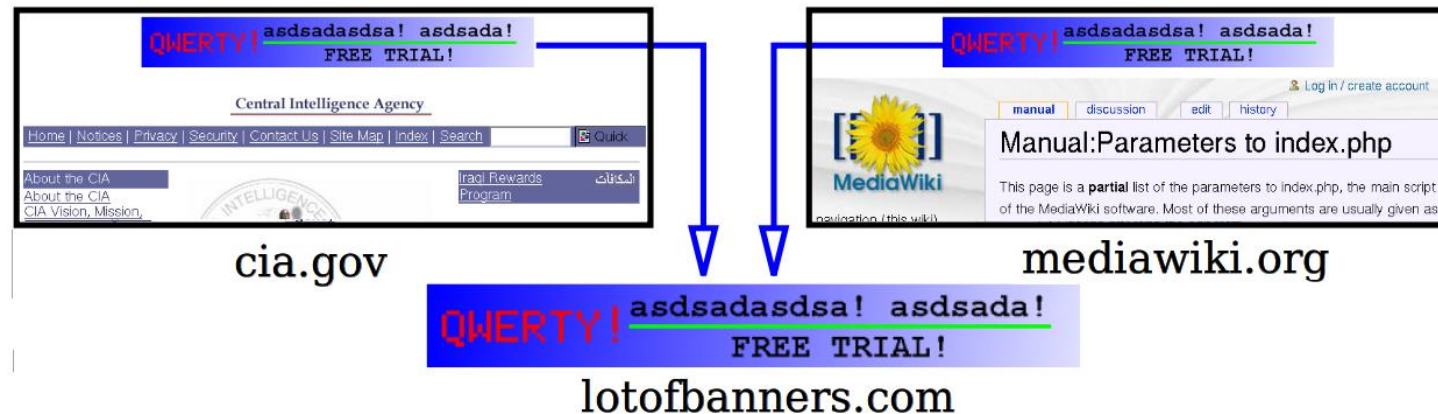
- **Myths:**

- Cookies are like worms/viruses and can erase data from the user's hard disk.
- Cookies are a form of spyware and can steal your personal information.
- Cookies generate popups and spam.
- Cookies are only used for advertising.

- **Facts:**

- Cookies are only data, not program code.
- Cookies cannot erase or read information from the user's computer.
- Cookies are usually anonymous (do not contain personal information).
- Cookies CAN be used to track your viewing habits on a particular site.

A "tracking cookie"



- An advertising company can put a cookie on your machine when you visit one site, and see it when you visit another site that also uses that advertising company.
- Therefore they can tell that the same person (you) visited both sites.
- Can be prevented by telling your browser not to accept "third-party cookies".

Cookie Components

- Here's an example of an HTTP header to create a cookie:
- Set-Cookie: fontSize=3; expires=Tuesday, 6-Jan-2019 17:53:08 GMT; path=/; domain=.example.com; HttpOnly

Setting a Cookie in PHP

- PHP provides a built-in function, `setcookie()` to send a cookie to the browser from PHP script.
- Examples:
- `setcookie("fontSize", 3, time() + 60 * 60 * 24 * 365, "/", ".example.com", false, true);`

Setting a Cookie in PHP

- `setcookie("pageViews", 7, 0, "/", "", false, true);`
- You can also update an existing cookie.
- You need to supply the **path and expires arguments when updating the cookie:** `setcookie("pageViews", 8, 0, "/", "", false, true);`

Accessing Cookies in Your Scripts

- The PHP `$_COOKIE` super global variable is used to retrieve a cookie value.
- So to display the pageViews cookie set in the previous example, you could use:
- `echo $_COOKIE["pageViews"];`

Removing Cookies

- Setting the cookie to FALSE erases it.
 - `setcookie("name", FALSE);`
 - `setcookie("CouponNumber", FALSE);`
- You can also set the cookie and pass in an expires argument that is in the past.
 - `setcookie("fontSize", "", time() - 3600, "/", ".example.com", false, true);`
 - This example sets the fontSize cookie's expiry time to one hour in the past, which effectively deletes it from the browser.

Sessions in PHP

- A session is a temporary set of variables that exists only until the browser has shut down.
- The idea of session control is to be able to track a user during a single session on a website.
- Session in PHP is driven by a unique session ID.
- All the session data is stored as keys and values pairs in the `$_SESSION` superglobal array.

Sessions in PHP

- Why do we need session?
- What are the pros of using Sessions over cookies?

Sessions in PHP

- `session_start()`- is used to start a session
- `$_SESSION['variable_name']`- is used to store data in session variable
- `session_destroy()`- is used to destroy a session
- `unset($_SESSION['variable_name'])`- is used to unset a specific variable

Passing variables using sessions

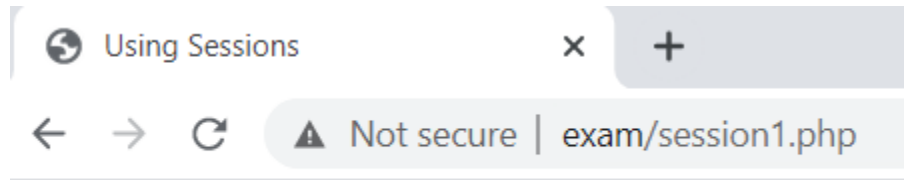
```
<?php
    session_start();
?>

<html>
<head>
    <title>Using Sessions</title>
</head>
<body>
    <?php
        $_SESSION['name'] = 'Ali';
    ?>
    <h1> Welcome to the first page </h1>
    <br>
    <a href = "session2.php"> Go to the next page</a>
</body>
</html>
```

Passing variables using sessions

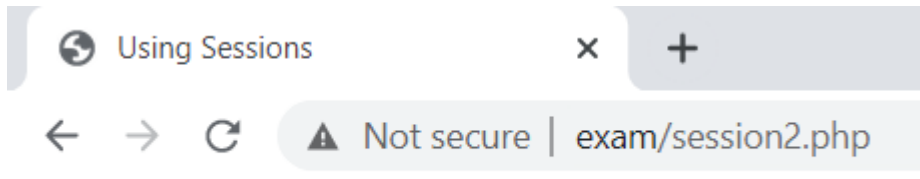
```
<?php
session_start();
?>
<html>
<head>
<title>Using Sessions</title>
</head>
<body>
<h1> This is the second page</h1>
<?php
echo "The value received from session variable is ". $_SESSION['name'];
?>
<br>
<a href = "session3.php"> Go to the third page</a>
</body>
</html>
```


Passing variables using sessions



Welcome to the first page

[Go to the next page](#)



This is the second page

The value received from session variable is Ali

[Go to third page](#)

Destroying a Session

- `session_destroy()` merely erases the session data from the disk.
- The data is still in the `$_SESSION` array until the current execution of the script ends.
- To make sure all session has been erased, initialize the `$_SESSION` array:
- `$_SESSION = array();`
- `session_destroy();`

Destroying a Session

- A trace of the session may still remain in the form of the PHPSESSID cookie in the user's browser.
- To make sure you have wiped the session from both the server and the browser, destroy the session cookie:

```
– if ( isset( $_COOKIE[session_name()] ) )  
  {  
    setcookie( session_name(), "", time()-3600, "/" );  
  }  
$_SESSION = array();  
session_destroy();
```

Destroying a Session

```
1  <?php
2      session_start();
3      $_SESSION = array();
4      session_destroy();
5  ?>
6  <HTML>
7  <HEAD>
8      <TITLE>Destroying Sessions</TITLE>
9  </HEAD>
10 <BODY>
11 <?php
12     echo $_SESSION['name'];
13 ?>
14 </BODY>
15 </HTML>
```

Destroying a Session

Destroying Sessions x +

← → ↻ ⚠ Not secure | exam/session3.php

(!) Notice: Undefined index: name in C:\wamp64\www\Exam\session3.php on line 11

Call Stack

#	Time	Memory	Function	Location
1	0.0002	405872	{main}()	...\session3.php:0

`$_Server` Super Global Variable

- Is used to get information from the server.
- `$_SERVER[" PHP_SELF "]`
- `$_SERVER[" SCRIPT_FILENAME "]`
- `$_SERVER["REMOTE_ADDR"]`
- `$_SERVER[" HTTP_USER_AGENT "]`
- `$_SERVER[" REQUEST_METHOD "]`
- `echo "Your IP address is: " .
$_SERVER["REMOTE_ADDR"]; // displays the IP
address of the visitor ' s computer (or proxy
server).`

`$_FILES:` super-global variable

- `$_FILES:` contains any item uploaded to the server when the post method is used.
- an array type variable
- Created automatically
- Can be accessed on other pages

`$_FILES`: super-global variable

- Keeps information about
- Name
- Size
- Type
- Tmp_name

\$_FILES: super-global variable

- FORM attributes required:
- Method should be post
- Enctype should be multipart/form-data
- `<form enctype="multipart/form-data" method="post" action="upload.php">`

\$_FILES: super-global variable

```
<body>
<form method="post"
  enctype="multipart/form-data"
  action="action.php">
  <input type="text" name="name">
  <input type="file" name="pic">
  <input type="submit">
</form></body>
```

File type

name

Asad

Choose File Mypci.jpg

submit

\$_POST

name

Asad

\$_FILES

[name] => file name [type] => file type [size] => file size [tmp_name] => tmp name

pic

`$_FILES`: super-global variable

- Accessing file information
 - `$_FILES['input-field name']['name'];`
 - `$_FILES['pic']['name'];`
 - `$_FILES['input-field name']['type'];`
 - `$_FILES['pic']['type'];`
 - `$_FILES['input-field name']['size'];`
 - `$_FILES['pic']['size'];`

Connecting PHP with MySQL.

- `mysqli_connect("hostname" ,"username",
"password")`
 - `mysqli_connect("localhost","root","")`
- `mysqli_select_db("database name")`
 - `mysqli_select_db("testdatabase")`

Inserting data in database

- Create form to receive input from user
- On action page
 - Retrieve user's input
 - Validate user's input (optional)
 - Establish connection with database
 - Write insert command
 - Execute command

Inserting data in database

- Insert SQL command:
- INSERT INTO `table_name` (list of columns) VALUES (list of values)
- INSERT INTO users ('user_Name','user_Email','user_Password') VALUES ('\$name','\$email','\$password')
- mysqli_query(query to execute)


\$_FILES: super-global variable

```
<table border="1" cellpadding="3" cellspacing="3">
<form name = "form1" enctype = "multipart/form-data" method = "post" action = "form_action_File.php">
<tr>
<th colspan="2">User Registration Form</th>
</tr>
```

```
<tr>
<td>Picture</td>
<td><input type = "file" name = "pic" value = "" /></td>
</tr>
```

\$_FILES: super-global variable

form_Action_File.php:

```
 <?php
$pic = $_FILES['pic'];
echo "<br>";
echo $_FILES['pic']['name']."<br>";
echo $_FILES['pic']['type']."<br>";
echo $_FILES['pic']['size']."<br>";
echo $_FILES['pic']['tmp_name']."<br>";
?>
```


Uploading file

- `move_uploaded_file()`:
- `move_uploaded_file (string $filename , string $destination);`
- If the file is valid, it will be moved to the filename given by destination.
- If the destination file already exists, it will be overwritten.

Uploading file

- File upload steps:
- Identify the file to be uploaded
 - tmp_name is used
- Define destination
 - Location + file name
- Upload the file

Uploading file

```
<?php
$filename = $_FILES['pic']['name'];
$tmpname = $_FILES['pic']['tmp_name'];

$destination = "images\\".$filename;
$con = mysqli_connect('localhost','root','','connections','3308') or die("could not connect to the server");
mysqli_select_db($con, 'connections') or die("could not select the db");
move_uploaded_file($tmpname, $destination);
$query = "INSERT INTO `uplaodimages`(`Images`) VALUES ('$filename')";
mysqli_query($con,$query);
echo "Upload Successful";
?>
```

Uploading file

- Restricting Users:
 - Size restriction
 - Type restriction
 - File rename

Uploading file

```
<?php

$filename = $_FILES['pic']['name'];
$tmpname = $_FILES['pic']['tmp_name'];
$size = $_FILES['pic']['size'];
$type = $_FILES['pic']['type'];
$con = mysqli_connect('localhost','root','','connections','3308') or die("could not connect to the server");
mysqli_select_db($con, 'connections') or die("could not select the db");

if($size<10000 AND $type = 'image/jpeg')
    move_uploaded_file($tmpname,"images/$filename");
else
    echo "Invalid File";
```

References

- Chapter 10, “Beginning PHP6,Apache,Mysql web development” by Matt Doyle, Wrox publishers, 2009, ISBN: 0470413964
- Chapter 9 “Beginning PHP6,Apache,Mysql web development” by Matt Doyle, Wrox publishers, 2009, ISBN: 0470413964