Built in functions in PHP 5

As we discussed functions in previous lectures, so in this lecture we will continue to discuss following functions.

* Session
* Cookie

# Session

A session is a mechanism to persist information across different web pages to identity users when they navigate from one page to another in a single site or app. Are you wandering why sessions are used for websites? To see why websites need session we need to know about HTTP protocol. HTTP is a stateless protocol as it is not responsible for storing information. For example if you navigate from one page to another, this protocol does not know who you are because when you access a web page the protocol is only responsible for providing the contents of the requested page, and if you navigate between the pages the protocol only provides the contents of the requested page as it is not related to previous page because the server interprets each request separately. There is no way to store information in the protocol. The following diagram depicts the HTTP protocol.

**Server**

**CLIENT**

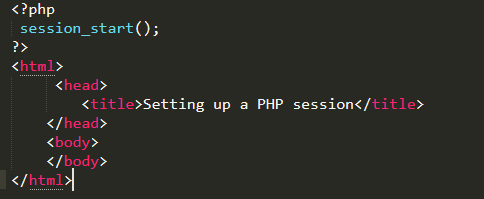
Request

Response

In the diagram client sends a request and server sends the response. If you want to display specific user information in each response, you have to authenticate the user name every time. Suppose you have to type your name on every page to let the server know who you are? it becomes a milestone. So sessions allows to store the information on different web pages thus it helps to maintain a state. This lets the server knows, that request are generating from the same user.

**How to start a session?**

A session is start by using **session\_start()** functionin your file. Session is placed at the top of file because session sends the information in HTTP headers so that the information would be send before any page content loads. The following code demonstrates how to do this.



Session is started, now we have to send variables in session function.

Session associative array allows us to store the data in it. For example if we want to store increment counter in session so that, when each time the user loads the page the counter is incremented until you the destroy session or the user close the browser. Now do this practically.

**Code:**

<?php

session\_start(); //session start

if( isset( $\_SESSION['counter'] ) ) { //check if session is set

$\_SESSION['counter'] += 1; //increment counter

}else {

$\_SESSION['counter'] = 1;

}

?>

<html>

<head>

<title>Setting up a PHP session</title>

</head>

<?php echo $\_SESSION['counter'] // display the counter?>

<body>

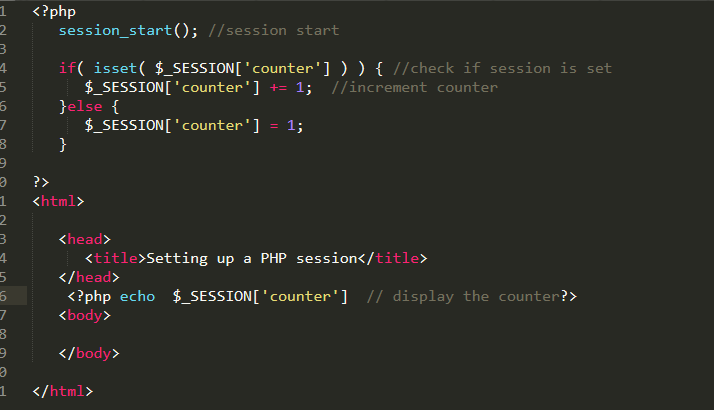
</body>

</html>

In the above example first of all session is start, then **if** condition is used with **isset** function to check whether session is set or not. Session variable is stored as an increment counter. Now each time when you open the file, it increment the counter.

Session generates a unique identifier for each user, so that when different users open file in different browsers it generate unique id which helps to identify the user.

Here is the pictorial illustration of the above code:



Figure

**How to destroy a session?**

A session can be destroyed by using (**session\_destroy**) function. This function doesn’t need any argument to destroy the **session**, just call the function it will destroy all the session variables.

# Cookies

A cookie is often used to identify a user. A cookie is a small file with a size of 4KB that the web server stores to client computer. It is used to keep tracking of user information such as user name and user password so that, when the user visits the website next time he does not need to provide the username and password again. A cookie is always set in the HTTP headers.

**Syntax**

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

**Parameters:** The session cookie require 6 parameters in general,

**Name:** It is used to set the name of cookie.

**Value:** It is used to set the value of cookie.

**Expire:** it is used to set the expiry timestamp to cookie after which the cookie can’t be accessible**.**

**Path:** It is used to specify the path of server to which cookie is available.

**Domain:** it is used to specify the domain

**Security**: It is used to check whether the HTTP connection is secure so that the cookie should be sent.

**Creating a cookie:**

**Code:**

<?php

$value="Olivia";

setcookie("name", $value, time()+ 60,'/'); // expires after 60 seconds

?>

Save the following code in another file read.php

<?php include('index.php');

if(!isset($\_COOKIE)) {

echo "Cookie name" . $value. "is not set!";

} else {

echo "Cookie not 'is set!<br>";

}

?>

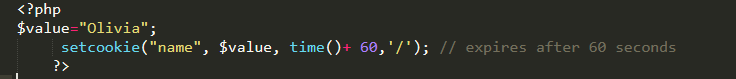
In the above example the cookie is set by providing name value and time that is equal to 60 seconds.

($value) variable contains a string in it. Now we check the other file by using isset function that whether the cookie is set or not.($\_COOKIE)is a super global array. If you open the index file then cookie is set and you can open read.php on your browser so you can get the value.

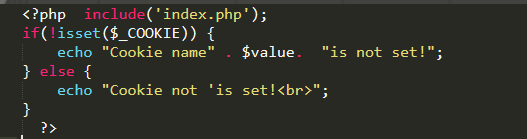
If you want to modify the cookie just change the value in the cookie.

If you want to delete the cookie just set the time in past with minus sign.

Here is the piece of above code in figure number 2 and 3.



Figure



Figure

**Difference between cookie and session**

The main difference between session and cookie is that cookies are stored on a user's computer in the text file format while sessions are stored on the server side.

Cookies can't hold multiple variables on the other hand session can hold multiple variables.

You can manually set an expiry for a cookie, while session only remains active as long as the browser is open.

|  |  |
| --- | --- |
| **Cookie** | **Session** |
| Client side file | Server side file |
| Remember info until its expiry date | Remembers info until website time out or user switch off the browser. |
| Usually contain id string | Usually contain complex information |
| An expiry date can be set to destroy cookie | No time limit can be set to destroy session |
|  |  |