Cookie and Session

Prof.N.Nalini

AP(Sr)

VIT

Cookies and Sessions

- Internet is based on Hypertext Transfer Protocol (HTTP), which is a stateless protocol.
- Maintaining the state between your subsequent visits to a Web page prevent loss of sensitive data
- We need to keep track of information about the web user/client between consecutive HTTP requests

  Example: Shopping Cart
  - “Remember Me”
  - For storing the items selected by the site users in their respective shopping carts.

Cookies

- Cookies are files stored on the client side
- Contains relevant data in name-value pairs
- Comes with expiration dates. Expired cookie contents are no longer accessed by the browser
- Can be managed by server-side scripts(PHP)
  - Relevant cookies are automatically submitted from client to server with HTTP request
  - PHP stores information in $_COOKIE superglobal

Use of Cookies

- To determine how many users visit the given Web site and how often
- For storing details of the users who visit the site or register on the Web site.
- Allowing users to customize the interface (such as layout and colors) as per their liking.
- To prevent repetitive logins, thus making the login process faster. In addition, since the cookie is stored at the client end, the Web server need not be burdened each time a user needs to log in to the site. The server only needs to authenticate the first-time users.
- For tracking a user's path and activities on a given Web site. This feature allows the Web administrators to track miscreants.
- For generating individual user profiles. For example, some sites display personalized messages to their users when they log in to the site.

Setcookie()

- To set a cookie in PHP:
  - **setcookie(name, value, expiration, path, domain, securemode)**
  - **setcookie(“mycookie”,”hello”, time()+60,”/”,”localhost”,0)**
- Time of expiry entered in seconds.
  - Present time + time in seconds until expiration[note:time()]
- Once set, S_COOKIE[‘**mycookie**’] will have value ‘**hello**’
- Always check with isset($_COOKIE[$cookie_name]) before trying to use the cookie’s value
- **Path** - This parameter is used to limit the scope of a cookie to a certain part of the document tree within the Web server.
- **Domain** - This parameter is used to specify the domain for which the cookie is valid.
- **Security parameter** – This parameter ensures that the confidential data stored in the cookie is safe from unauthorized access while it travels from the Web server to the client machine
- To delete a cookie, set a new cookie with same arguments but expiration in the past

Sample cookies program

1. Setcookie.php

```
<?php
$username = “aabbcc”;
setcookie(‘username’, $username, time() + 60 * 60 * 24); // cookie for 1 day
echo $_COOKIE[‘username’] .” created with expiration time of 1 day”;
setcookie(‘secondcookie’,$username); //expiration time is Session time.
setcookie(‘third’,’sample’,time()+60*60*24*30);//cookie for 30 days
//accessing cookie
echo $secondcookie;
?>
```

Sample cookies program

TestCookie.php

```
<html>  <head>
<title>Beginning PHP, Apache, MySQL Web Development</title>
</head> <body>
<h1>This is the Test Cookie Page</h1>
<?php
if ($_COOKIE[‘username’] == “” || $_COOKIE[‘password’] == “”)
{
?>
No cookies were set.<br>
<a href=”setcookie.php”>Click here</a> to set your cookies.
<?php }
else
{  ?>
Your cookies were set:<br>
Username cookie value: <b><?php echo $_COOKIE[‘username’]; ?></b><br>
Password cookie value: <b><?php echo $_COOKIE[‘password’]; ?></b><br>
<?php
} ?>  </body>  </html>
```

Page Count program

```
<?php
if (!isset($_COOKIE['kookie']))  {
        $pagecount=0;
        setcookie("kookie",$pagecount);
        echo "<font size=8><center>This is the first time u have accessed
this page<br>";
        echo "<center>A cookie was sent to u & stored in ur
computer<br>";
        }
else    {
        $pagecount = ++$_COOKIE['kookie'];
        setcookie("kookie",$pagecount);
        echo "<center><font size=10 color = red> view count:
".$_COOKIE['kookie'];
        }
?> <html><body>
<b> <center> <font size=8 color=blue> Refresh button will refresh the page
and the page count</b></center> </body> </html>
```

Delete Cookie program

```
<?php
echo "<font size=8>";
if (isset($_COOKIE["kookie"]))
   {
    setcookie("kookie","",time()-10); // deletes cookie
    echo "Cookie named as <color = red>kookie </color>was
deleted";
   }
else
 echo "there is no cookie with name 'KOOKIE'";
?>
```

Visitor Count program

```
<?php
$visitor_ip = $_COOKIE["user_ip"];
$counter = "counter.txt";
$counter_file_line = file($counter);
if(!$vistor_ip) {
  setcookie("user_ip", $REMOTE_ADDR, time()+360000);
  $counter_file_line[0]++;
  $cf = fopen($counter, "w+");
  fputs($cf, "$counter_file_line[0]");
  fclose($cf);
} elseif($vistor_ip != $REMOTE_ADDR) {
  $counter_file_line[0]++;
  $cf = fopen($counter, "w+");
  fputs($cf, "$counter_file_line[0]");
  fclose($cf);
} ?>
```

Sessions

- data stored on the server, managed by Server-side script(PHP)
- In PHP, session variables store information about user session in $_SESSION superglobal array.
- Session variables hold information about one single user, and are available to all pages in one application.
- Session variables expire when the browser is closed

PHP Session management

session_start()

- Before you can store user information in your PHP session, you must first start up the session.
- The `session_start()` function must appear at the top of EVERY page, BEFORE the <html> tag

```
<?php
ini_set('session.save_path','c:/wamp/www/session');
$_SESSION['views']=1;


if(isset($_SESSION['views']))
  $_SESSION['views']=$_SESSION['views']+1;
else
  $_SESSION['views']=1;
echo "Views=". $_SESSION['views'];


?>
```

Session

ses1.php

```
<?php
ini_set('session.save_path','c:/wamp/www/session');
        session_start();
       $s=session_save_path();
        echo $s;
        if (!isset($_SESSION['name']))
        {
        $_SESSION['name'] = "hello";
         echo $_SESSION['name']."<br>";
         }
         session_destroy();

       ?>
```

```
<html>
   <head><title>Starting a Session</title></head>
<body><a href="ses2.php">next page</a>
            </body>
            </html>
```

ses2.php

```
<?php
            session_start();
            if(isset($_SESSION['name']))
           echo "name:".$_SESSION['name'];
           else
           echo "session expires";
              ?>
```

PHP Session management

- `unset():`
  - function used to free the specified session variable
  - Example: `unset($_SESSION[‘shopping_cart']);`

- `session_destroy();`
  - Resets current session. You will lose all your stored session data.
  - Call when a user signs out

Working of a Session

Sessions work by creating a **unique identification(UID**) number for each visitor and storing variables based on this ID. This helps to prevent two users' data from getting confused with one another when visiting the same webpage.

- Session_start() – it is required, for every session's program.
- $_SESSION['session_name'] – can access the created sessions through this super global.
- isset($_SESSION['session_name']) – used to check the availability of a session.
- unset($_SESSION['session_name']) – deletes the session
- Session_destroy() – deletes all sessions.

Session Functions

- **session_start** — Initialize session data.
- **session_destroy** — Destroys all data registered to a session
- **session_unset** — Free a session
- **session_name** — Get and/or set the current session name
- **session_register** — Register one or more global variables with the current session.
- **session_destroy** — Destroys all data registered to a session.
- **session_id** — Get and/or set the current session id.
- **Session_regenerate_id** — Update the current session id with a newly generated one
- **session_is_registered** — Find out whether a global variable is registered in a session