Working with Cookies

Session 16

Objectives

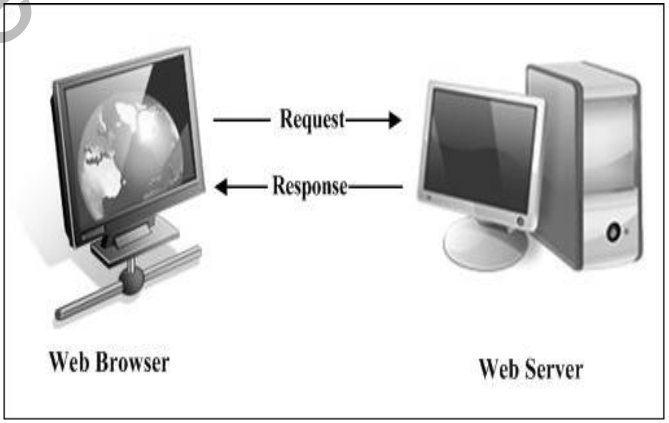
- *Describe the process of setting a cookie*
- *Explain the process of retrieving a cookie in PHP*
- *Explain the process to delete a cookie*
- *Identify the drawbacks associated with cookies*

Introduction

- Web sites store user information in databases to maintain a track of their visits
- Cookies enable Web sites to store user information
- PHP supports Hyper Text Transfer Protocol (HTTP) cookies

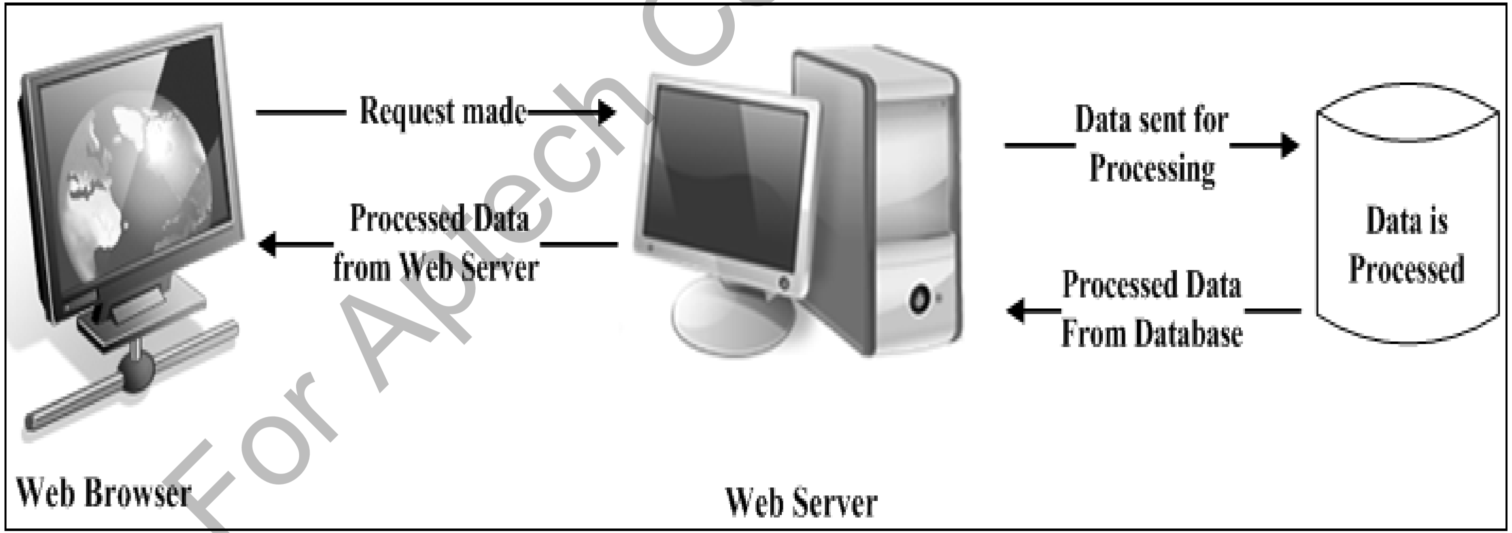
Web Pages

- Uses HTTP protocol for sending information to the server
  - HTTP is a stateless protocol, because the execution of the current command is completed without the knowledge of commands that came before it
- Are of two types:
  - Static Web pages
  - Dynamic Web pages
- Static Web pages
  - Web browser requests for a page and the server completes the request by sending the required file
  - Does not involve any interaction with the user



Web Pages 2-2

- Dynamic Web pages
  - Require user interaction, so scripting languages such as JavaScript, PHP, and ASP are used
  - Accept information from the user and record it for further processing



Introducing a Cookie

- Data stored by Web sites are as follows:
  - Temporary information is stored in cookies for a stipulated period
  - Permanent data is stored in cookies for a certain period and then the required information is saved in the database
- Types of cookies are as follows:
  - **Persistent** - exist in the Web browser for a period specified at the time of its creation
  - **Non-persistent** - deleted from the Web browser as soon as the user exits the browser

Introducing a Cookie

- Web sites use cookies to determine the following:
  - Number of times the user has visited the Web site
  - Number of new visitors
  - Number of regular users
  - Frequency of a user visiting the Web site
  - Date on which the user had last visited the Web site
  - Customized Web page settings for a user

Working of a Cookie

- When a user visits the Web site for the first time, the Web server creates a unique ID and sends the ID in the cookie to the Web browser
- Browser stores the cookie and sends it back to the Web site in subsequent requests
- Life of a cookie depends on the expiration time and date
- Cookie is stored on the hard disk of the user's computer which enables the Web site to keep a track on the user visiting the Web site
- Web servers and Web browsers send cookies to each other in HTTP headers
- Web server sends the cookie to the browser in the `setcookie` header field which is part of the HTTP response
- Web browser stores the cookie and uses the same in subsequent requests to the same Web server

Working of a Cookie

- Consider the following HTTP response header:

Snippet

```
HTTP/2.0 200

Content-Length: 8451

Content-Type: text/html

Date: Mon, 27 Dec 2010 05:29:24 GMT

Expires: Mon, 27 Dec 2010 05:29:44 GMT

setcookie: city=east-coast-usa
```

Working of a Cookie

- In the code, the following information is displayed:
  - Version number of the HTTP protocol
  - Size of the content
  - Type of the content
  - Date and time of response
  - Expiry date and time of the cookie
  - Cookie header

Working of a Cookie

- Consider the following HTTP response header:

Snippet

```
GET /usa/florida.php HTTP/2.0

Connection: Keep-Alive

Cookie: city=east-coast-usa

Host: www.Webworldmaps.com

Referrer: http://www.Webworldmaps.com/
```

Cookie can be defined using the `setcookie` function.

Code displays a subsequent request that the Web browser sends to the Web server.

Setting a Cookie

- Setting a cookie is sending the cookie to the browser
- PHP uses two functions, `setcookie()` and `setrawcookie()` to set a cookie
- `setrawcookie()` function sends a cookie without encoding the cookie value
- `setcookie()` function generates the cookie header field that is sent along with the rest of the header information

Setting a Cookie

- The `setcookie()` function is as follows:

Syntax

```
setcookie(name, value, expiry date, path, domain, secure)
```

Where,

name - defines the name of the cookie

value - defines the value of the cookie that is stored on the client system

expiry date - defines the date and time (UNIX timestamp) when the cookie will expire

path - defines the location on the server where the cookie will be stored.

domain - defines the domain name where the cookie is made available

secure - defines the type of HTTP connection that the cookies will pass through

Setting a Cookie

- When the cookie is set, the value is automatically encoded in the URL
- When the script retrieves a cookie, it automatically decodes the value from the URL
- Cookies are a part of the HTTP header and there can be more than one cookie in the header, but it should relate to the same domain or Web site
- The code related to the cookies must be specified before the following:
    - HTTP header
    - Displaying any content
    - Any white space
- If any content is displayed before calling the `setcookie()` function, the function will fail and return `False`
- If the `setcookie()` function runs successfully, the function returns `True`

Setting a Cookie

- Setting a cookie that expires in one day in a Web site that displays country maps when a user enters a country name in the search feature of the Web site are as follows:

Snippet

```
$mapname = $_GET['fmapname'];

setcookie("mycookie", $mapname, time()+86400,
"/Webmap/", ".Webworldmaps.com");
```

Setting a Cookie

- In the code, `fmapname` is the variable that contains the country name that the user enters
- The `$mapname` variable stores the value that the GET method retrieves from the form
- The `setcookie()` function includes the following:
  - `mycookie` - defines the name of the cookie
  - `time()+86400` - specifies the time when the cookie will expire
  - `/Webmap` - defines the location where the cookie will be stored
  - `.Webworldmaps.com` - specifies the domain that the cookie will use

Setting a Cookie

- Creating a cookie that expires when the Web browser window is closed are as follows:

Snippet

```
$val = $_GET['uname'];
setcookie("uname",$val);
```

In code, `uname` is the variable that contains a value.
The `$val` variable stores the value of `uname` that the `GET` method retrieves.
The `setcookie()` function in the code snippet sets a cookie named `uname`.
The value of `$val` is assigned to the cookie, `uname`.

Retrieving Cookies in PHP

- Cookies are useful only when the Web server can retrieve the information from it
- The Web browser matches the URL against a list of all the cookies present on the client system
- If the Web browser finds a match, a line containing the name value pairs of the matched cookie is included in the HTTP header
- Document that created the cookie as well as that are present in the same directory can access it
- Documents outside the directory need to include the path or the domain name of the cookie to access the cookie

Retrieving Cookies in PHP

- PHP provides three ways of retrieving a cookie value and they are as follows:
  - Passing a variable as the cookie name - retrieve the cookie value, use the variable as the cookie name. The following code snippet displays a cookie value:

Snippet

```
echo $cookie_name;
```

  - This method of retrieving the cookie value is not recommended as PHP will start searching all the variables present in the client system
  - The `register_globals` option must be enabled in the configuration file

Retrieving Cookies in PHP

- Using `$_COOKIE` array
  - PHP uses cookie name as a variable to retrieve the cookie value. PHP can also use an associative array called `$_COOKIE` to retrieve the cookie value
  - The `$_COOKIE` is a global variable that reads a value of the cookie
  - An example of this is shown as follows:

Snippet

```
echo $_COOKIE [$cookie_name];
```

  - This is more reliable and faster than retrieving the cookie value through a variable

Retrieving Cookies in PHP

Snippet

```
<?php

$cookieval = $_COOKIE ['uname'];   ?>


<HTML>

<BODY>

<?php

if (isset($cookieval))

{

echo "Welcome $cookieval";

}
```

Retrieving Cookies in PHP

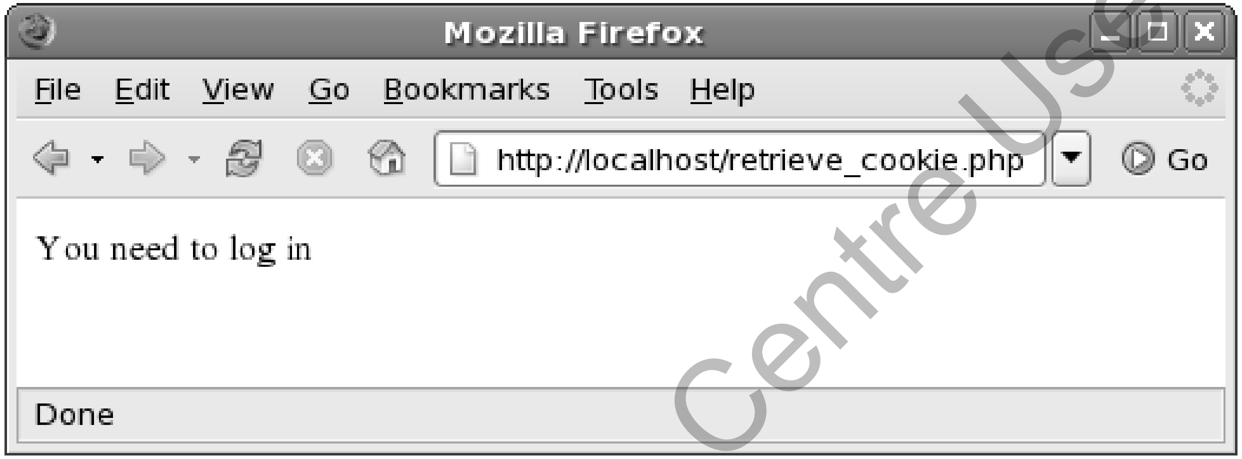
- **`retrieve_cookie.php`** - Retrieving a cookie value using the $_COOKIE global variable

Snippet

```
else

{

echo "You need to log in";

}

?>

</BODY>

</HTML>
```

Retrieving Cookies in PHP

The output of the script is as follows:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/retrieve_cookie.php
Go
Done

```
You need to log in
```

`$cookieval` stores the cookie value.

The `isset()` function checks whether the cookie is set

Deleting a Cookie

- Cookies can be deleted automatically or manually
- There are two ways to delete a cookie, which are as follows:
  - Resetting the expiry time of the cookie to a time in the past
  - Resetting a cookie by specifying the name of the cookie
- When you create a cookie that has the same name and time as an existing cookie, the existing cookie is deleted from the hard drive of the client
- To delete a cookie with a date in the past, enter the code as shown in the Snippet in a PHP script

Snippet

```
setcookie("$cookie_name", "", time()-8000);
```

- In the code, `$cookie_name` refers to the name of the cookie. The value of the cookie is not specified and the `time()` function accepts the expiration date in the past

Deleting a Cookie

- This process is called as deconstructing the variable
- Use the following syntax to delete a cookie through deconstruction:

Snippet

```
setcookie($cookie_name);
```

- To delete the cookie named **`uname`**

Snippet

```
setcookie($uname);
```

Problems with Cookies

- Web sites store user-related information on the client system
- Cookies are not secure and reliable because the user-related information can be accessed by anyone who has full access to the client system
- Following are some of the drawbacks of cookies:
  - Cookies cannot contain more than a certain amount of information
  - Only a maximum of 29 cookies of a domain can be maintained
  - A browser can maintain maximum of 300 cookies
  - Storing large number of cookie files slows down the system
  - Some users disable cookies while accessing Web sites as a result Web sites that depend on cookies lose information of such users

Problems with Cookies 2-2

- There can be multiple users using the same system visiting the same Web site
- Web sites assign cookies to the system and not to the user. This can hamper the number of visitor's statistics
- A cookie can contain large amount of information and retrieving larger amount of information on each page requires repetitive coding across the pages

Summary

- Web sites use cookies, stored on the hard disk of the client system, to store user-specific information
- Dynamic Web pages gets information from the user and records it for further processing
- Persistent cookies are stored in the Web browser for a period specified during the time of its creation and non-persistent cookies are deleted from the Web browser as soon as the user exits the browser
- The HTTP header, transmitted between the Web server and the Web browser, contains cookies
- A cookie can be retrieved by passing a variable as a cookie name and using the $_COOKIE[] variable

Summary

- PHP uses the setcookie() and setrawcookie() functions to set a cookie
- The two ways to delete a cookie are resetting the expiry time of the cookie to a time in the past and by resetting the cookie by specifying the name of the cookie
- The maximum number of cookies that can be maintained for a domain is 20. A browser can maintain maximum of 300 cookies