PHP - Cookies

The worldwide web is powered by HTTP protocol, which is a stateless protocol. The mechanism of Cookies helps the server maintain the information of previous requests. PHP transparently supports HTTP cookies.

- When a client first sends its request, the server includes a small piece of data along with its response as cookies. PHP provides the **setcookie()** method to inject cookies in the response.
- This cookie data is stored in the client's machine as text files. On subsequent visits of the same client, these cookies are included as a part of the **request** header.
- The server populates the PHP superglobal variable "$_COOKIE" with all the cookies present in the client request.

This chapter will teach you how to set cookies, how to access them and how to delete them.

The Anatomy of a Cookie

Cookies are usually set in an HTTP header (although JavaScript can also set a cookie directly on a browser). A PHP script that sets a cookie might send headers that look something like this −

```
HTTP/1.1 200 OK
Date: Fri, 04 Feb 2000 21:03:38 GMT
Server: Apache/1.3.9 (UNIX) PHP/4.0b3
Set-Cookie: name=xyz; expires=Friday, 04-Feb-07 22:03:38 GMT;
                 path=/; domain=tutorialspoint.com
Connection: close
Content-Type: text/html
```

As you can see, the Set-Cookie header contains a name value pair, a GMT date, a path and a domain. The name and value will be URL encoded. The expires field is an instruction to the browser to "forget" the cookie after the given time and date.

If the browser is configured to store cookies, it will then keep this information until the expiry date. If the user points the browser at any page that matches the path and domain of the cookie, it will resend the cookie to the server.The browser's headers might lo something like this −

```
GET / HTTP/1.0
Connection: Keep-Alive
User-Agent: Mozilla/4.6 (X11; I; Linux 2.2.6-15apmac ppc)
Host: zink.demon.co.uk:1126
Accept: image/gif, */*
Accept-Encoding: gzip
Accept-Language: en
Accept-Charset: iso-8859-1,*,utf-8
Cookie: name=xyz
```

A PHP script will then have access to the cookie in the environmental variables $_COOKIE or $HTTP_COOKIE_VARS[] which holds all cookie names and values. Above cookie can be accessed using $HTTP_COOKIE_VARS["name"].

How to Set a Cookie in PHP?

PHP contains the **setcookie** function to create a cookie object to be sent to the client along with HTTP response.

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

Parameters

Here is the detail of all the arguments −

- **Name** − This sets the name of the cookie and is stored in an environment variable called HTTP_COOKIE_VARS. This variable is used while accessing cookies.
- **Value** − This sets the value of the named variable and is the content that you actually want to store.
- **Expiry** − This specify a future time in seconds since 00:00:00 GMT on 1st Jan 1970. After this time cookie will become inaccessible. If this parameter is not set then cookie will automatically expire when the Web Browser is closed.
- **Path** − This specifies the directories for which the cookie is valid. A single forward slash character permits the cookie to be valid for all directories.
- **Domain** − This can be used to specify the domain name in very large domains and must contain at least two periods to be valid. All cookies are only valid for the host and domain which created them.
- **Security** − This can be set to 1 to specify that the cookie should only be sent by secure transmission using HTTPS otherwise set to 0 which mean cookie can be sent by regular HTTP.

Example

The PHP script give below checks if the cookie named **username** is already set, and retrieves its value, if so. If not, a new cookie **username** is set.

```
<?php
   if (isset($_COOKIE['username'])) {
      echo "<h2>Cookie username already set:" . $_COOKIE['username'] . "</h2>";
   } else {
      setcookie("username", "MohanKumar");
      echo "<h2>Cookie username is now set</h2>";
   }
?>
```

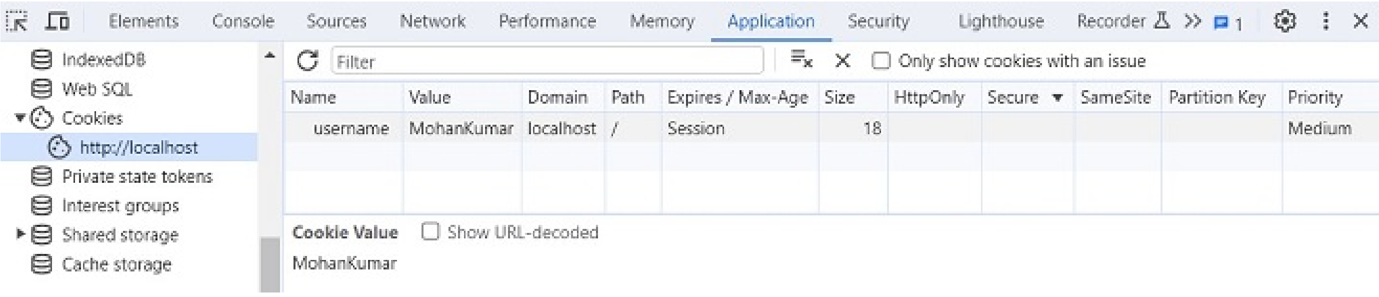
Run this script from the document root of the Apache server. You should see this message –

```
Cookie username is now set
```

If this script is re-executed, the cookie is now already set.

```
Cookie username already set: MohanKumar
```

Your browser’s developer tool is a very useful facility. You can set, retrieve and delete cookies with its help. The cookie set by the above program can be viewed under the Application tab of the browser’s developer tools.



Name	Value	Domain	Path	Expires / Max-Age	Size	HttpOnly	Secure	SameSite	Partition Key	Priority
username	MohanKumar	localhost	/	Session	18					Medium

Cookie Value □ Show URL-decoded

MohanKumar

A **foreach** loop as below retrieves all the cookies –

```
<?php
   $arr=$_COOKIE;
   foreach ($arr as $key=>$val);
   echo "<h2>$key=>$val </h2>";
?>
```

The following script contains an HTML form. It sends the form data to setcookie.php script, that sets the cookies with the use of data retrieved from the **$_POST** array.

The HTML form is rendered by the following code −

```
<form action="setcookie.php" method="POST">
   <input type="text" name="name">
   <input type="text" name="age">
   <input type="submit" name="Submit">
</form>
```

SetCookie.php reads the form data and sets the cookies.

```
if (isset($_POST["submit"]) {
   setcookie("name", $_POST["name"]);
   setcookie("age", $_POST["age"]);
}
```

With another getcookie.php code, we can retrieve the cookies set.

```
if (isset($_COOKIE["name"])
echo "Cookie: name => " . $_COOKIE["name"]. "<br>";
if (isset($_COOKIE["age"])
echo "Cookie: age => " . $_COOKIE["age"]. "<br>";
```

Accessing Cookies with PHP

PHP provides many ways to access cookies. Simplest way is to use either $_COOKIE or $HTTP_COOKIE_VARS variables. Following example will access all the cookies set in above example.

```
<?php
   echo $_COOKIE["name"]. "<br />";

   /* is equivalent to */
   echo $HTTP_COOKIE_VARS["name"]. "<br />";

   echo $_COOKIE["age"] . "<br />";
   /* is equivalent to */
   echo $HTTP_COOKIE_VARS["age"] . "<br />";
?>
```

You can use **isset()** function to check if a cookie is set or not.

```
<?php
   if( isset($_COOKIE["name"]))
      echo "Welcome " . $_COOKIE["name"] . "<br />";

   else
      echo "Sorry... Not recognized" . "<br />";
?>
```

Deleting the Cookies

To delete cookie set the cookie with a date that has already expired, so that the browser triggers cookie removal mechanism.

Example

Take a look at the following example −

```
<?php
   setcookie("username", "", time() - 3600);
   echo "<h2>Cookie username is now removed</h2>";
?>
```

The browser shows the following response −

```
Cookie username is now removed
```

You may also set array cookies by using array notation in the cookie name.

```
setcookie("user[three]", "Guest");
setcookie("user[two]", "user");
setcookie("user[one]", "admin");
```

If the cookie name contains dots (.), PHP replaces them with underscores (_).

Although the main purpose behind the concept of cookies is to help web developers provide a more personalized and convenient user experience, it may pose a risk to your privacy and personal information.

In some cases, the application may deny you full access you don't accept their cookies. In such cases, periodically clearing the cookie related data from your browser’s cache is advised.