A cookie is a piece of information set by an HTTP server and sent to the web client within an HTTP response for storage by the client.

Cookies are associated with a particular website, and each time a client requests a page from a website for which it has received and stored a cookie, it includes the corresponding cookie's data within its HTTP request

HTTP is a stateless protocol meaning that each HTTP transaction has no memory of any previous transactions.

The purpose of cookies is to allow state, or knowledge of previous events to be introduced into HTTP transactions.

A cookie can hold state information:

- Identify a particular client

- Contain information from a previous HTTP transaction such as shopping cart contents

In the header of the HTTP response you include:

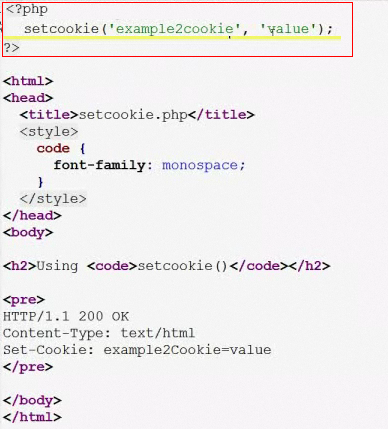
Set-Cookie: cookie=value

In the HTTP request header

Cookie: cookie=value

In PHP you set the cookie before any script

**SIMPLE COOKIE DECLARATION**



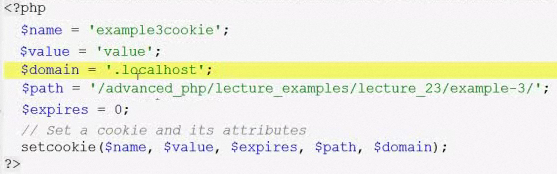
**WITH ATTRIBUTES**

The most common attributes

- domain: the domain which a cookie applies to

- path: path within a domain that the cookie applies to

- expires: until when is active, if expires is "0" then the cookie expires when the browser is closed

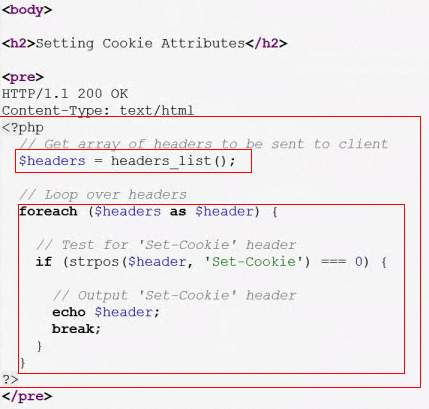


$expires = time() + 60 \* 60; // it will expire in 1 hour

To deactivate cookie set $expires = time() - 10; // set past time

**RETRIEVING COOKIE**

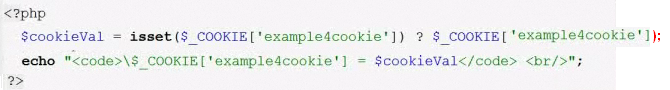
IMPORTANT: The cookie is created after you make ONE response from a server, it will not work in the same page at the first time.



PHP provides the $\_COOKIE superglobal that is an associate array containing

any cookie/name pairs

$cookies = $\_COOKIE['Cookie\_Name'];



**SECURITY**

Cookie can be used to identify and track users so they have privacy issues associated with them

Web users can disable cookies from being set on their computer

The implication is that if your application uses cookies some users may not be able to access any cookie

**SESSIONS**

- Client session if data is stored in the client

- Server session if data is stored in the server

The session data is stored

- Client: in a file, RAM

- Server: in a file, database, RAM

A client passes a Session ID to the server with each HTTP request via either:

- Cookies

- URL parameter (query string)

Session data is stored by PHP in flat files in the directory specified in session.save\_path

By default xampp\tmp

session\_start() is a built-in PHP function that when called either creates

a new session or continues an existing session

If a new session is created, session\_start() will generate an SID and send a cookie to

the client named session.name with its value set to the SID generated

If a session is being continued, session\_start() will look for a cookie or URL parameter named session.name

to laod an SID from.

Session\_start() must be called before any output is generated

**item.php**

<?php

class **Item** {

public $itemID;

public $name;

public $price;

public $description;

public $imageFileExt;

}

?>

<**body**>

<**h2**>Instantiate an Item Object</**h2**>

<?php

Include\_once ‘./classes/Item.php’;

$item1 = new Item();

$itemID = $item2->itemID;

$name = $item1->name;

$price = $item1->price;

$description = $item1->description;

$imageFileExt = $item1->imageFileExt;

?>

</**body**>

= = = = = = = = = = = = = = = = = = = = =

A cookie is created with the setcookie() function.

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

The following example creates a cookie with PHP. The cookie is named "user" and the value will be "John Doe". It will expire after 30 days (86400 \* 30). Using "/", means that the cookie is available in all website (otherwise, select the directory you prefer):

Example

<?php  
$cookie\_name = "user";  
$cookie\_value = "John Doe";  
setcookie($cookie\_name, $cookie\_value, time() + (86400 \* 30), "/");  
// 86400 = 1 day 3600 = 1 hora  
?>  
<html>  
<body>  
  
<?php  
echo "Cookie is set.";  
?>  
  
</body>  
</html>

|  |  |
| --- | --- |
| **Note** | **Note:** The setcookie() function must appear BEFORE the <html> tag. |

**Note:**The value of the cookie is automatically URLencoded when sending the cookie, and automatically decoded when received (to prevent URLencoding, use setrawcookie() instead).

Retrieve a Cookie Value

The PHP global variable **$\_COOKIE** is used to retrieve a cookie value.

<html>  
<body>  
  
<?php  
$cookie\_name = "user";  
if(!isset($\_COOKIE[$cookie\_name])) {  
    echo "Cookie named '" . $cookie\_name . "' does not exist!";  
} else {  
    echo "Cookie is named: " . $cookie\_name . "<br>Value is: " . $\_COOKIE[$cookie\_name];  
}  
?>  
  
</body>  
</html>

Modify a Cookie Value

To modify a cookie, just set (again) the cookie using the setcookie() function:

<?php  
$cookie\_name = "user";  
$cookie\_value = "Alex Porter";  
setcookie($cookie\_name, $cookie\_value, time() + (86400 \* 30), "/");  
// 86400 = 1 day  
?>  
<html>  
<body>  
  
<?php  
$cookie\_name = "user";  
if(!isset($\_COOKIE[$cookie\_name])) {  
    echo "Cookie named '" . $cookie\_name . "' does not exist!";  
} else {  
    echo "Cookie is named: " . $cookie\_name . "<br>Value is: " . $\_COOKIE[$cookie\_name];  
}  
?>  
  
</body>  
</html>

Delete a Cookie

To delete a cookie, use the setcookie() function with an expiration date in the past:

<?php  
$cookie\_name = "Login";  
unset($\_COOKIE[$cookie\_name]);  
// empty value and expiration one hour before  
$res = setcookie($cookie\_name, '', time() - 3600);  
?>

<html>  
<body>  
  
<?php  
echo "Cookie 'user' is deleted.";  
?>  
  
</body>  
</html>

Check if Cookies are Enabled

The following example creates a small script that checks whether cookies are enabled. First, try to create a test cookie with the setcookie() function, then count the $\_COOKIE array variable:

Example

<?php  
setcookie("test\_cookie", "test", time() + 3600, '/');  
?>  
<html>  
<body>  
  
<?php  
if(count($\_COOKIE) > 0) {  
    echo "Cookies are enabled";  
} else {  
    echo "Cookies are disabled";  
}  
?>  
  
</body>  
</html>