

What is a Cookie?

A cookie is often used to identify a user.

A cookie is a small file that the server embeds on the user's computer.

Each time the same computer requests a page with a browser, it will send the cookie too.

With PHP, you can both create and retrieve cookie values.

How to Create a Cookie? - Syntax

The `setcookie()` function is used to set a cookie.

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

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

How to Create a Cookie? - Example

In the example below, we will create a cookie named "user" and assign the value "Alex Porter" to it.

We also specify that the cookie should expire after one hour:

```
<?php
setcookie("user", "Alex Porter", time()+3600);
?>

<html>
<body>

</body>
</html>
```

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

How to Retrieve a Cookie Value?

The PHP `$_COOKIE` variable is used to retrieve a cookie value.

In the example below, we retrieve the value of the cookie named "user" and display it on a page:

```
<?php
// Print a cookie
echo $_COOKIE["user"];

// A way to view all cookies
print_r($_COOKIE);
?>
```

How to Retrieve a Cookie Value?

In the following example we use the `isset()` function to find out if a cookie has been set:

```
<html>
<body>

<?php
if (isset($_COOKIE["user"]))
    echo "Welcome " . $_COOKIE["user"] . "!<br />";
else
    echo "Welcome guest!<br />";
?>

</body>
</html>
```

How to Delete a Cookie?

When deleting a cookie you should assure that the expiration date is in the past.

Delete example:

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time()-3600);
?>
```

PHP Session Variables

When you are working with an application, you open it, do some changes and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end. But on the internet there is one problem: the web server does not know who you are and what you do because the HTTP address doesn't maintain state.

A PHP session solves this problem by allowing you to store user information on the server for later use (i.e. username, shopping items, etc). However, session information is temporary and will be deleted after the user has left the website. If you need a permanent storage you may want to store the data in a database.

Sessions work by creating a unique id (UID) for each visitor and store variables based on this UID. The UID is either stored in a cookie or is propagated in the URL.

Starting a PHP Session

Before you can store user information in your PHP session, you must first start up the session.

Note: The `session_start()` function must appear BEFORE the `<html>` tag:

```
<?php session_start(); ?>

<html>
<body>

</body>
</html>
```

The code above will register the user's session with the server, allow you to start saving user information, and assign a UID for that user's session.

Storing a Session Variable

The correct way to store and retrieve session variables is to use the PHP `$_SESSION` variable:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>

<html>
<body>

<?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>

</body>
</html>
```

Output:

```
Pageviews=1
```

Storing a Session Variable

The correct way to store and retrieve session variables is to use the PHP `$_SESSION` variable:

```
<?php
session_start();
// store session data
$_SESSION['views']=1;
?>

<html>
<body>

<?php
//retrieve session data
echo "Pageviews=". $_SESSION['views'];
?>

</body>
</html>
```

Output:

```
Pageviews=1
```

Storing a Session Variable

In the example below, we create a simple page-views counter.

The `isset()` function checks if the "views" variable has already been set. If "views" has been set, we can increment our counter. If "views" doesn't exist, we create a "views" variable, and set it to 1:

```
<?php  
  
session_start();  
  
if(isset($_SESSION['views']))  
    $_SESSION['views']=$_SESSION['views']+1;  
  
else  
    $_SESSION['views']=1;  
echo "Views=". $_SESSION['views'];  
?>
```

Destroying a Session

If you wish to delete some session data, you can use the `unset()` or the `session_destroy()` function.

The `unset()` function is used to free the specified session variable:

```
<?php
unset($_SESSION['views']);
?>
```

You can also completely destroy the session by calling the `session_destroy()` function:

```
<?php
session_destroy();
?>
```

Note: `session_destroy()` will reset your session and you will lose all your stored session data.