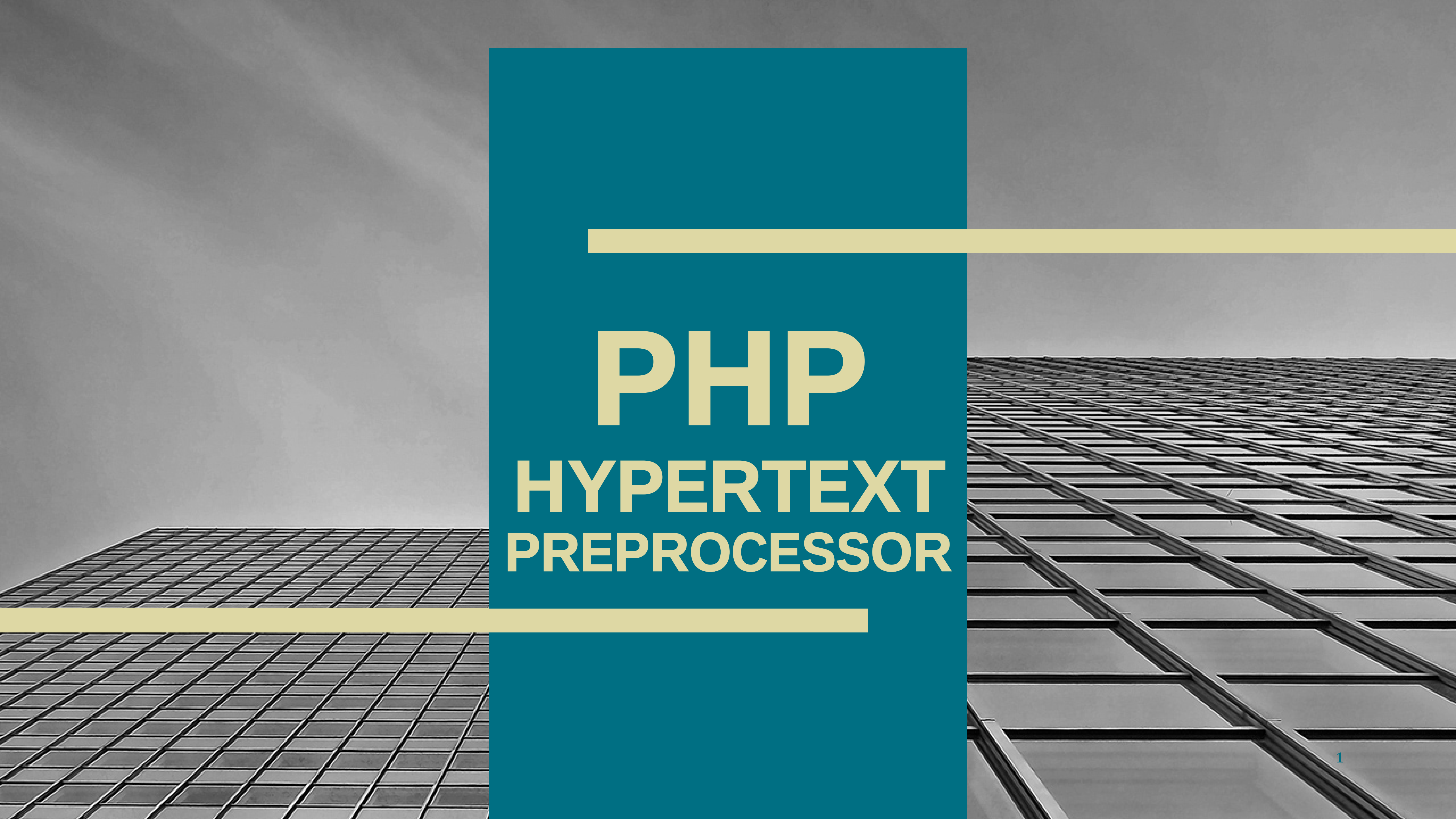


PHP

HYPERTEXT PREPROCESSOR

PHP

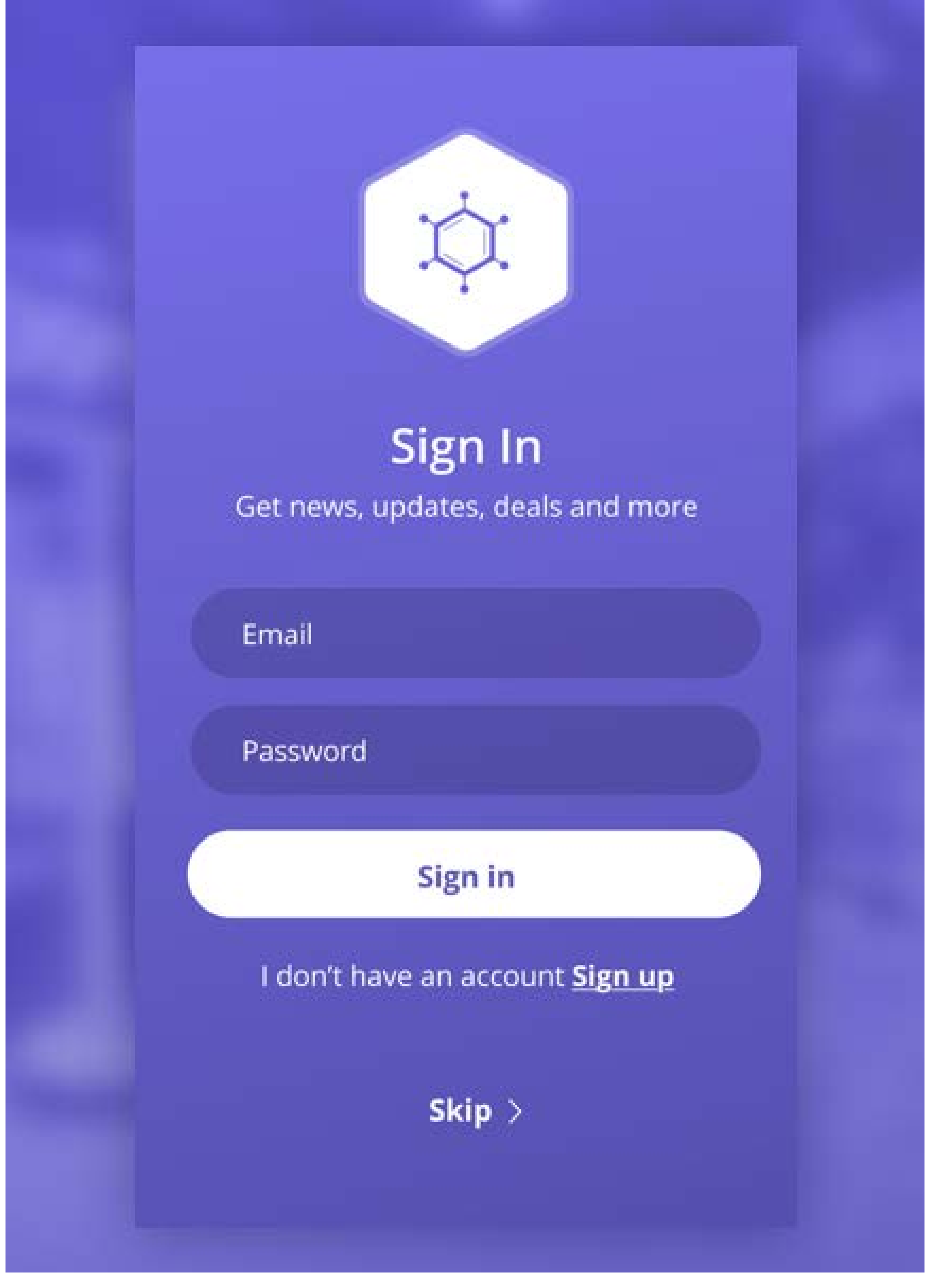
INTRODUCTION

We call it back-end programming language or we might call it a server-side programming language.

WHAT IS IT USED FOR?

You can make your website more dynamic.

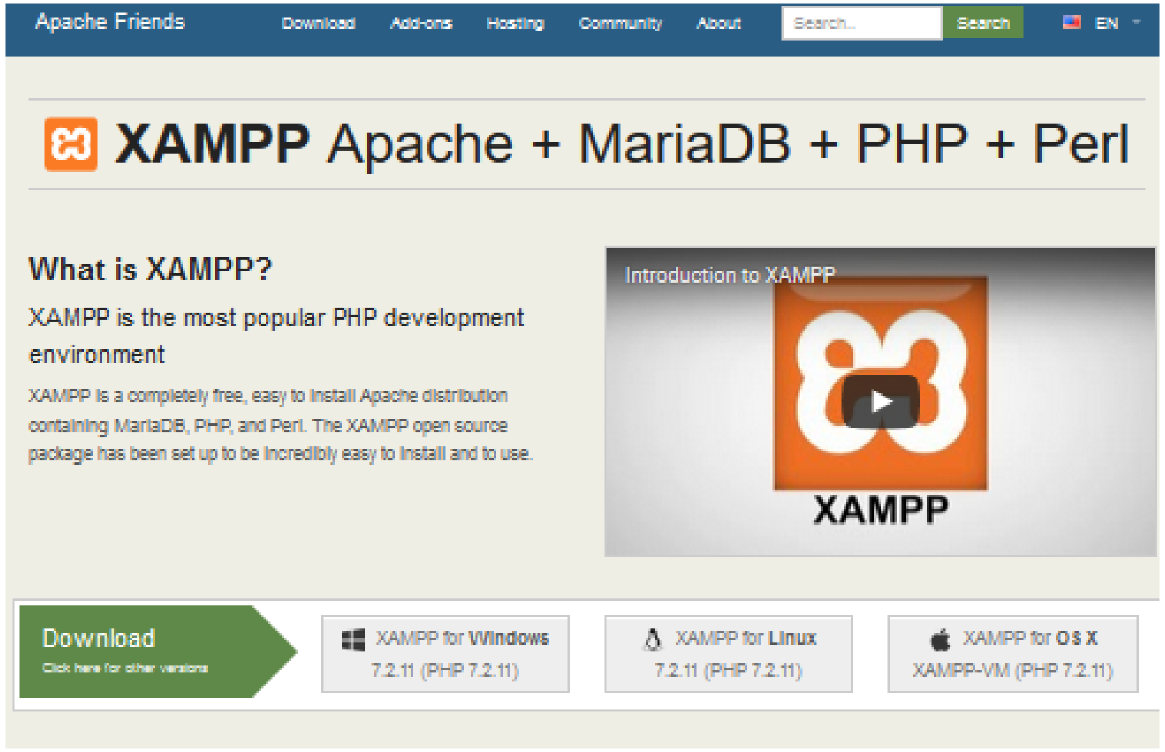
- As we learned at few chapters before about HTML, CSS, you could create a website where you can let users click on couple of links and read your text/article or maybe watch a video, some images.
- If you want to make it more useful, dynamic and users can actually do something on your website.
- For example: we have a log in system and people need their user accounts for sign up and with that accounts when they do log in and interact with the website such as posting something. Therefore, we need a database(storing or posting) for saving the information. To connect a website to the database(we will learn at the next few chapter), you will need the server-side language and we are going to focus on PHP.


Sign In
Get news, updates, deals and more
Email
Password
Sign in
I don't have an account Sign up
Skip

Source img: dribbble

INSTALLING A LOCAL SERVER: XAMPP

Go into https://www.apachefriends.org/index.html and choose your OS to download



Apache Friends
Download
Add-ons
Hosting
Community
About
Search...
Search
EN
XAMPP Apache + MariaDB + PHP + Perl

What is XAMPP?

XAMPP is the most popular PHP development environment

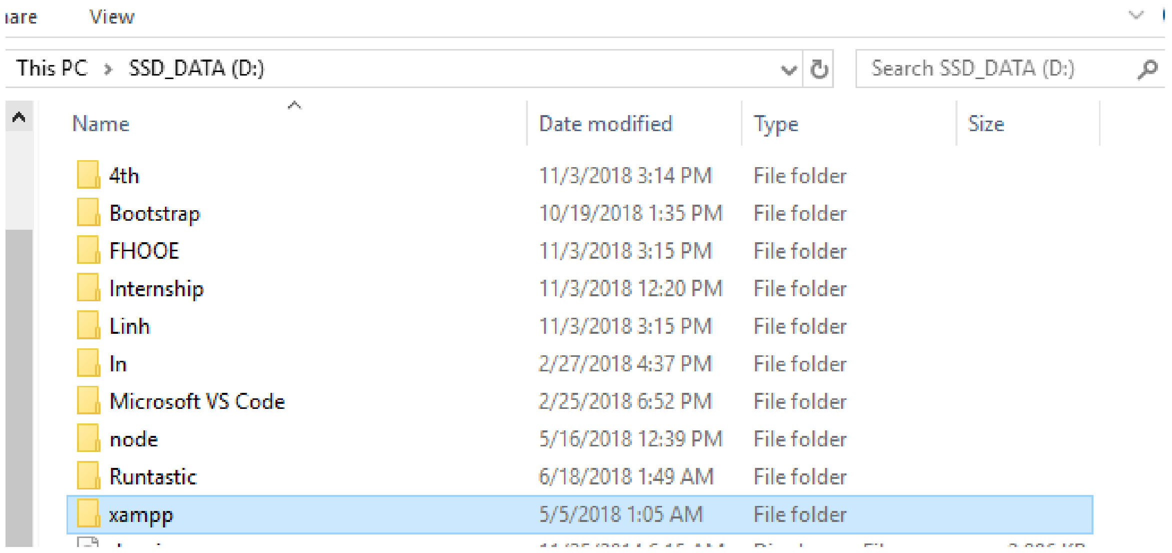
XAMPP is a completely free, easy to install Apache distribution containing MariaDB, PHP, and Perl. The XAMPP open source package has been set up to be incredibly easy to install and to use.

Introduction to XAMPP
XAMPP
Download
Click here for other versions
XAMPP for Windows 7.2.11 (PHP 7.2.11)
XAMPP for Linux 7.2.11 (PHP 7.2.11)
XAMPP for OS X XAMPP-VM (PHP 7.2.11)

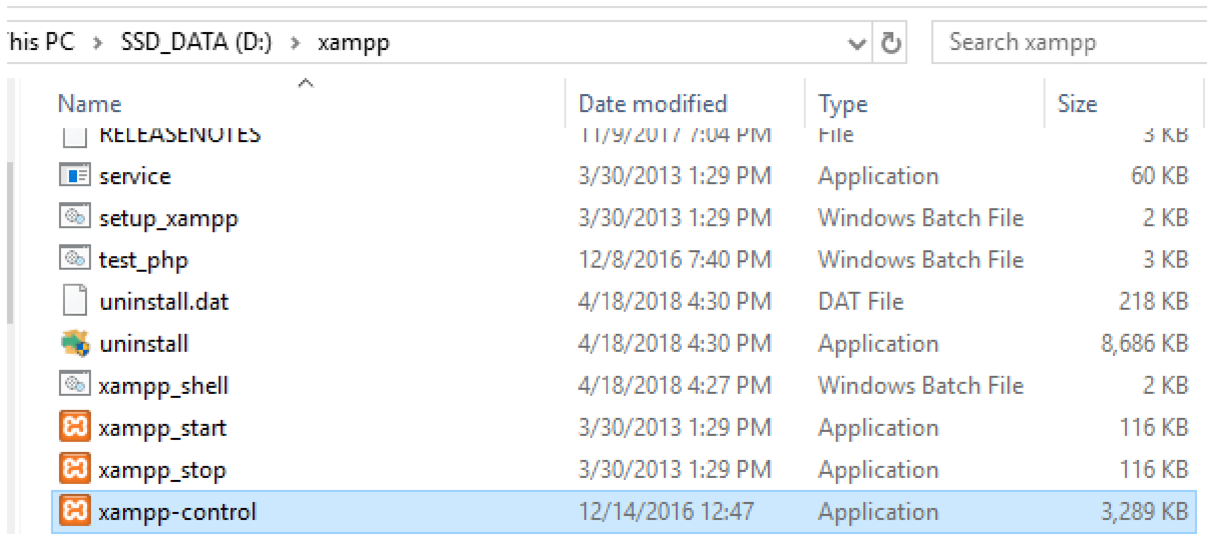
INSTALLING A LOCAL SERVER: XAMPP

Open the xampp downloaded folder that you may saved it somewhere in your disk drive.

Search and choose the xampp-control file.



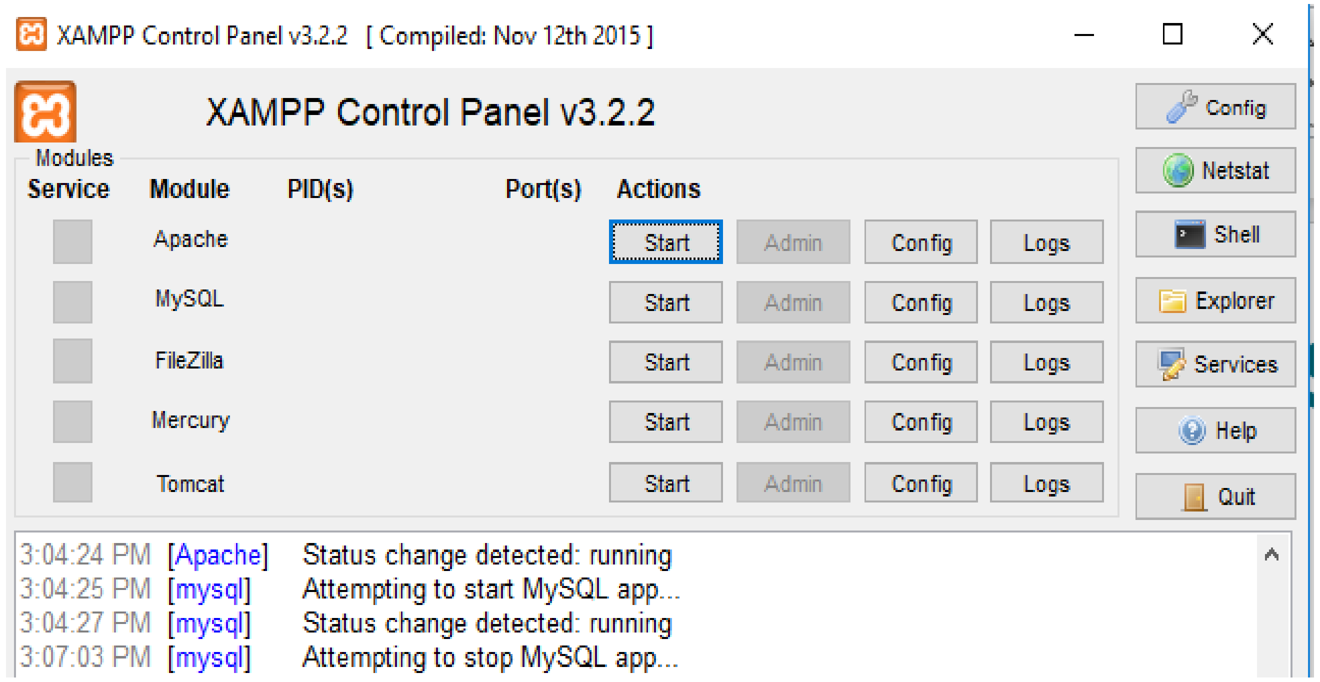
Name	Date modified	Type	Size
4th	11/3/2018 3:14 PM	File folder	
Bootstrap	10/19/2018 1:35 PM	File folder	
FHOOE	11/3/2018 3:15 PM	File folder	
Internship	11/3/2018 12:20 PM	File folder	
Linh	11/3/2018 3:15 PM	File folder	
In	2/27/2018 4:37 PM	File folder	
Microsoft VS Code	2/25/2018 6:52 PM	File folder	
node	5/16/2018 12:39 PM	File folder	
Runtastic	6/18/2018 1:49 AM	File folder	
xampp	5/5/2018 1:05 AM	File folder	



Name	Date modified	Type	Size
RELEASENOTES	11/9/2017 7:04 PM	File	3 KB
service	3/30/2013 1:29 PM	Application	60 KB
setup_xampp	3/30/2013 1:29 PM	Windows Batch File	2 KB
test_php	12/8/2016 7:40 PM	Windows Batch File	3 KB
uninstall.dat	4/18/2018 4:30 PM	DAT File	218 KB
uninstall	4/18/2018 4:30 PM	Application	8,686 KB
xampp_shell	4/18/2018 4:27 PM	Windows Batch File	2 KB
xampp_start	3/30/2013 1:29 PM	Application	116 KB
xampp_stop	3/30/2013 1:29 PM	Application	116 KB
xampp-control	12/14/2016 12:47	Application	3,289 KB

INSTALLING A LOCAL SERVER: XAMPP

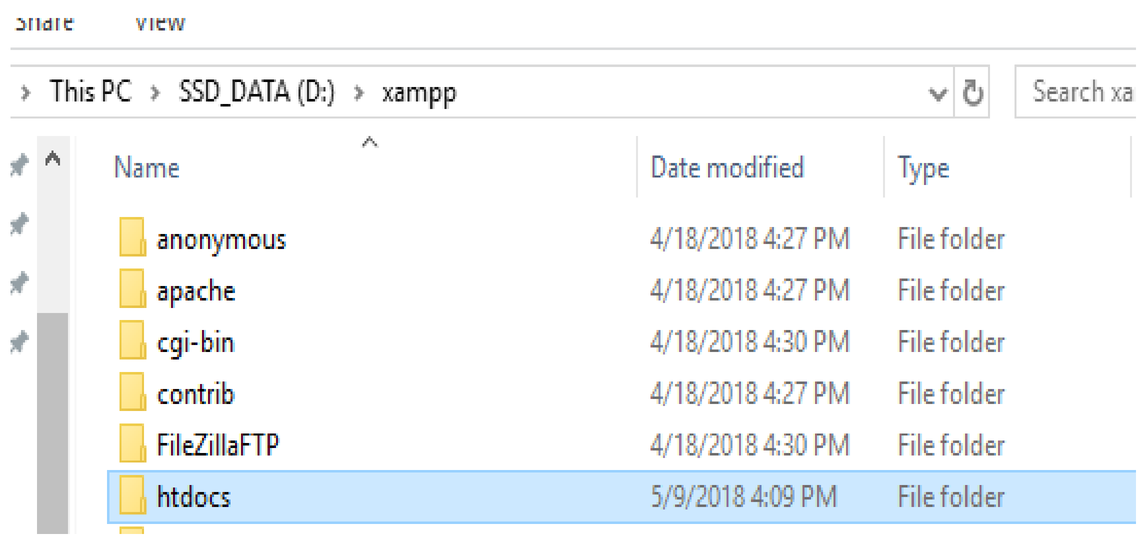
You could see the XAMPP Control Panel, click Start for Apache module and MySQL Module **(it lets the PHP code running/working)**



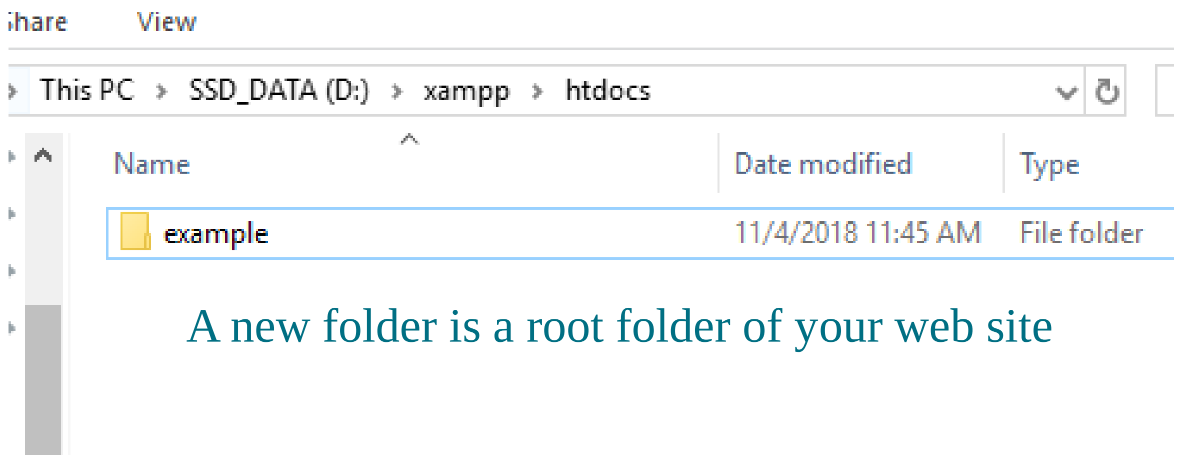
```
3:04:24 PM  [Apache]   Status change detected: running
3:04:25 PM  [mysql]    Attempting to start MySQL app...
3:04:27 PM  [mysql]    Status change detected: running
3:07:03 PM  [mysql]    Attempting to stop MySQL app...
```

INSTALLING A LOCAL SERVER: XAMPP

Go back to the xampp folder u did install before, look for and open the **htdocs** folder, there have some random default files which xampp installed, we should delete them all and create our new one as **example**



Name	Date modified	Type
anonymous	4/18/2018 4:27 PM	File folder
apache	4/18/2018 4:27 PM	File folder
cgi-bin	4/18/2018 4:30 PM	File folder
contrib	4/18/2018 4:27 PM	File folder
FileZillaFTP	4/18/2018 4:30 PM	File folder
htdocs	5/9/2018 4:09 PM	File folder

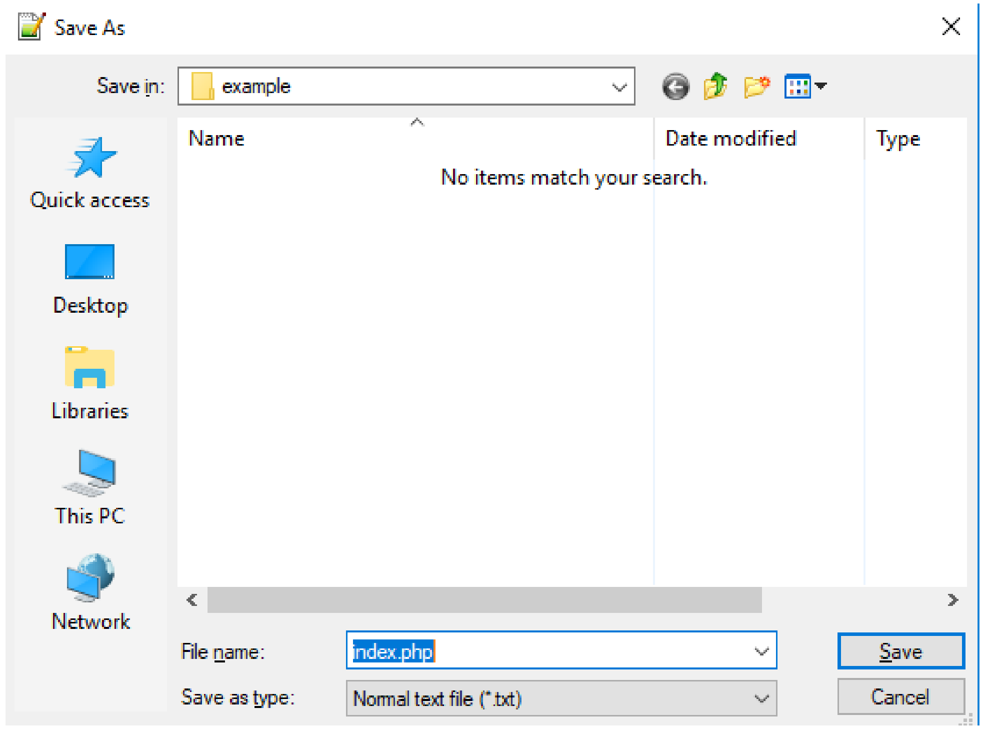


Name	Date modified	Type
example	11/4/2018 11:45 AM	File folder

A new folder is a root folder of your web site

HOW DOES IT WORK WITH A LOCAL SERVER

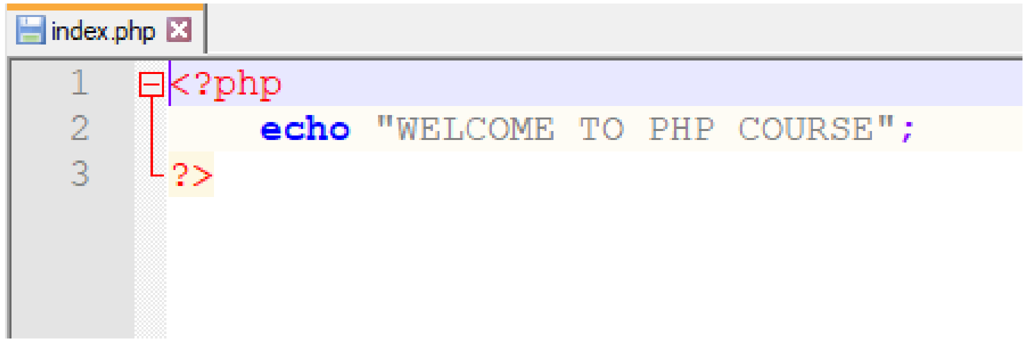
Create an index file and save as a php extension, the file has to be located inside the example folder that we just created.



No items match your search.

File name: index.php

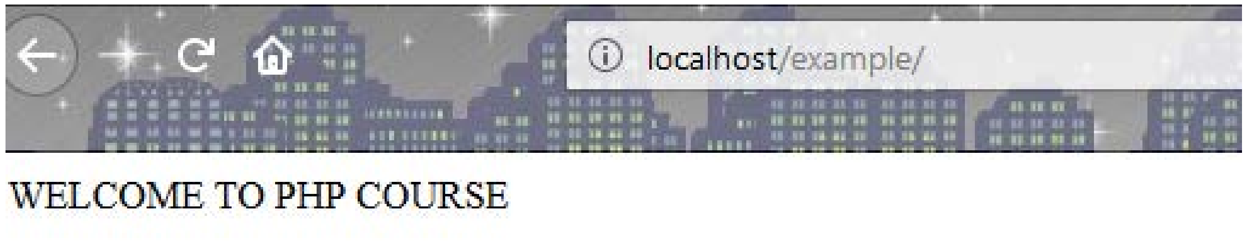
Save as type: Normal text file (*.txt)



```
<?php
    echo "WELCOME TO PHP COURSE";
?>
```

HOW DOES IT WORK WITH A LOCAL SERVER

Open up the Google Chrome browser and type in the URL **bar localhost/example** and you will see it works like a charm and of course it’s only running when you Start your Apache and MySQL

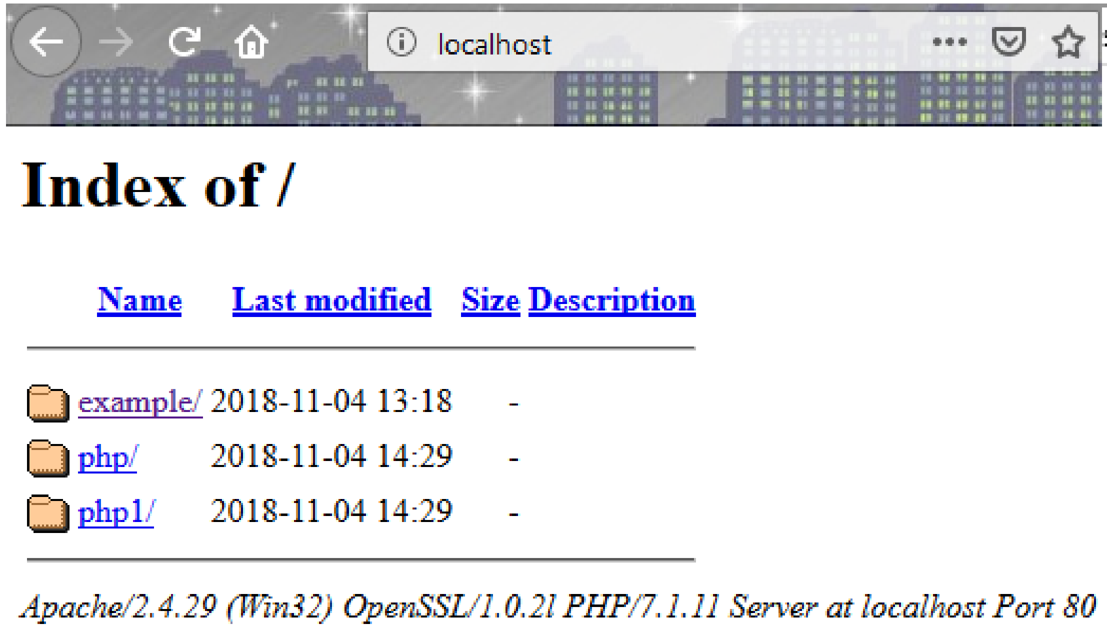


WELCOME TO PHP COURSE

You can Stop the XAMPP Control Panel and refresh the browser to see what happen.

HOW DOES IT WORK WITH A LOCAL SERVER

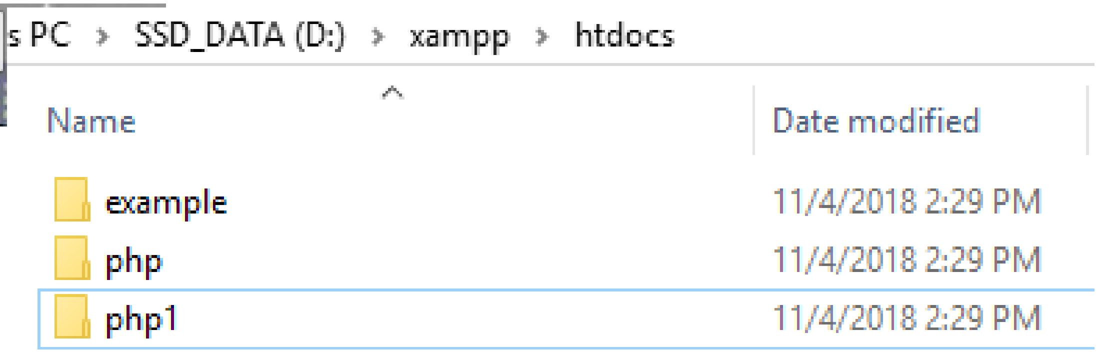
One more tip when you want to open up your php file,you only need to type **localhost** in your URL bar; from here you can see your example folder that u created before in your htdocs folder, you can also easily access to it and open the php file.


localhost

Index of /

```
Name        Last modified      Size Description
example/    2018-11-04 13:18    -
php/        2018-11-04 14:29    -
php1/       2018-11-04 14:29    -
```

Apache/2.4.29 (Win32) OpenSSL/1.0.2l PHP/7.1.11 Server at localhost Port 80


s PC › SSD_DATA (D:) › xampp › htdocs

Name	Date modified
example	11/4/2018 2:29 PM
php	11/4/2018 2:29 PM
php1	11/4/2018 2:29 PM

It has the same folders as I have in the localhost, so if you do not want to write anything behind localhost, you can use this way.

```
<?php
        Code goes here
?>
```

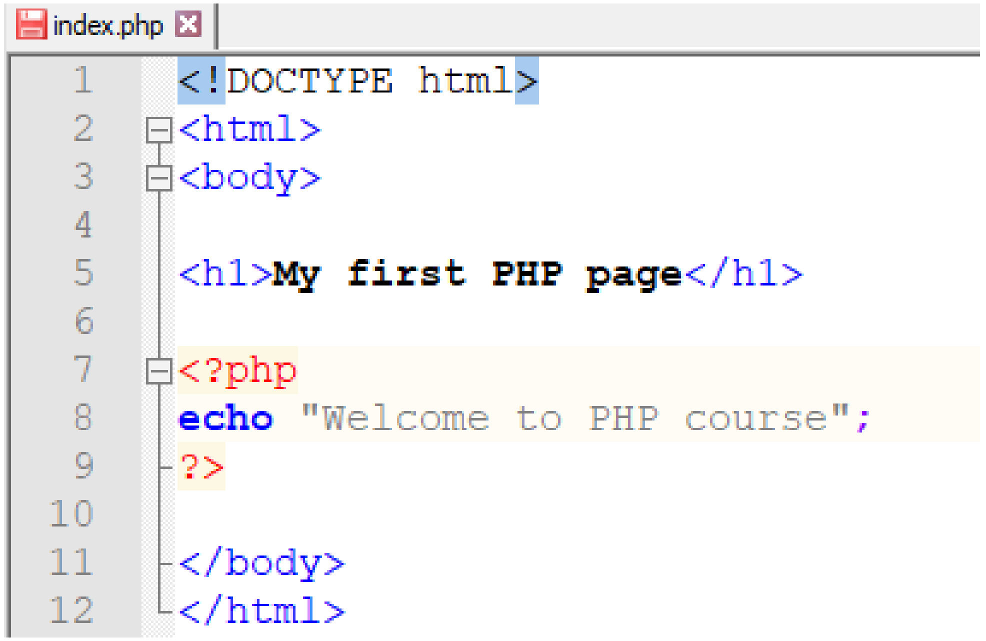
BASIC PHP SYNTAX: with HTML

When building a complex pages, it is necessary to combine PHP and HTML.

PHP is designed to interact with HTML by being included in HTML pages.

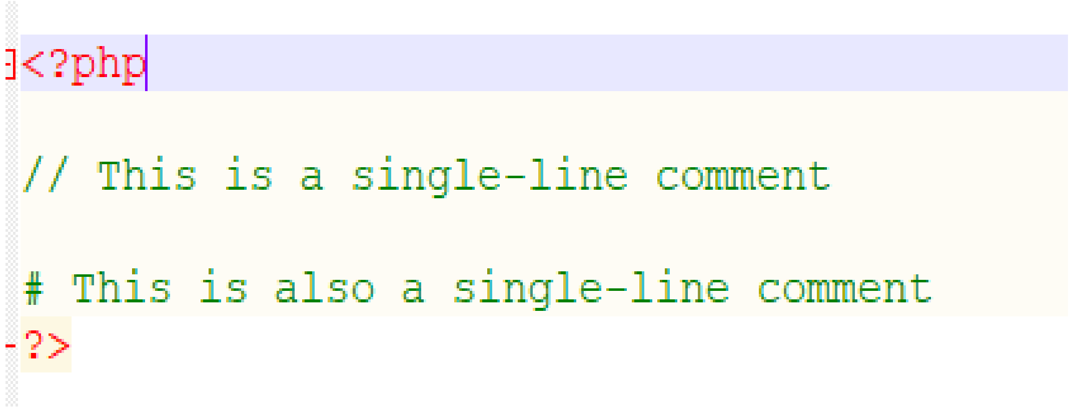
When a given file contains PHP code, it must have a PHP extension (.php)

Remember to run in the server!!!


index.php

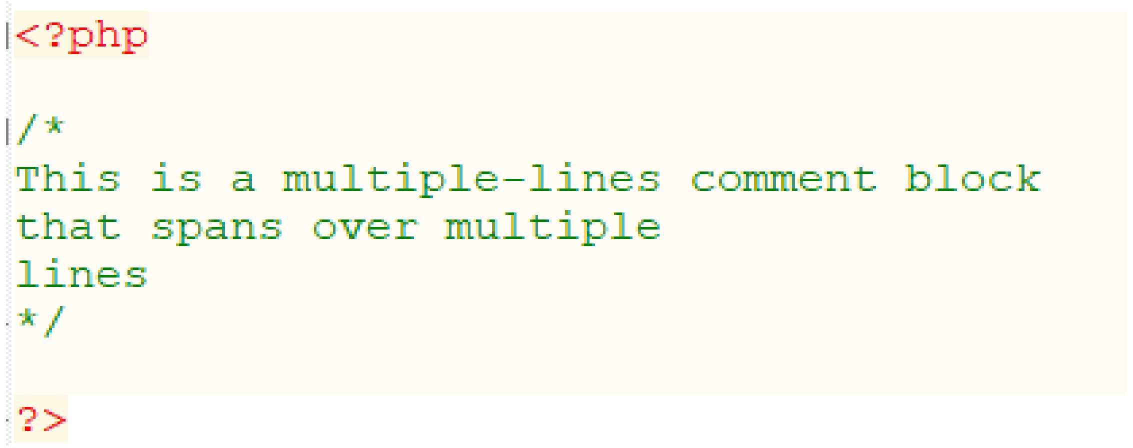
```
<!DOCTYPE html>
<html>
<body>

<h1>My first PHP page</h1>

<?php
echo "Welcome to PHP course";
?>

</body>
</html>
```

COMMENT IN PHP



```
<?php

// This is a single-line comment

# This is also a single-line comment
?>
```

Single line comment



```
<?php

/*
This is a multiple-lines comment block
that spans over multiple
lines
*/

?>
```

Multiple-lines comment

PHP VARIABLES

In PHP, a variable starts with the $ sign, followed by the name of the variable:

Variable syntax example

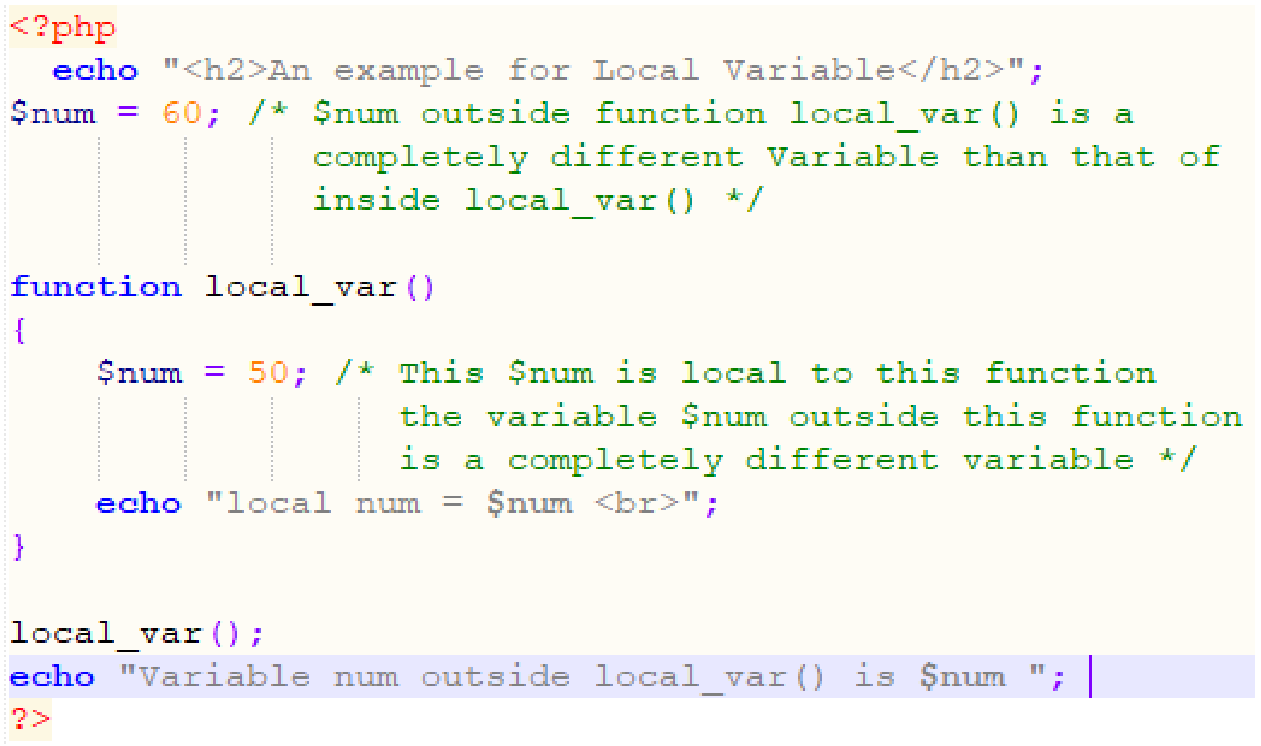
```
$name =“Kauko”;
$_hobby = “reading”;
$myAge = “30”;
```

PHP VARIABLE RULES

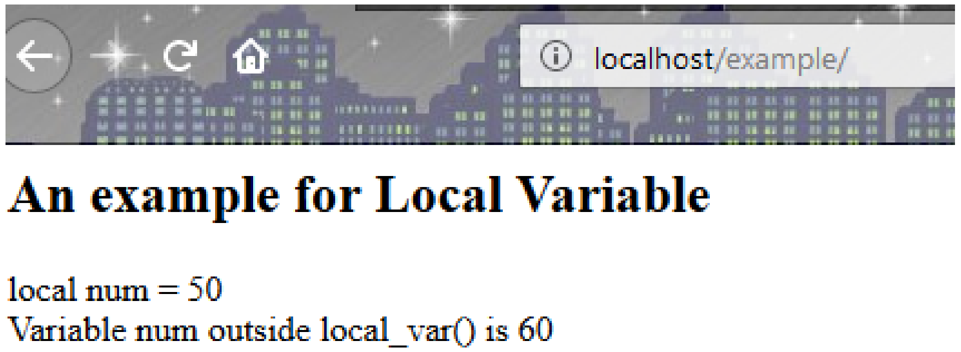
PHP variables start with the dollar sign, followed by the name of it: $variable
PHP variables must begin with a letter or underscore.
PHP variables can include alpha-numeric characters (letters and numbers) and underscores "_".
A variable name cannot contain spaces and cannot start with number.
Variable names are Case Sensitive so you should take care when defining variable names. $myAge and $MyAge are two different variables.

PHP LOCAL VARIABLE

Local variables are declared ***inside*** a function or blocks and it has its scope only in that particular function. It cannot be accessed outside that function.



```
<?php
  echo "<h2>An example for Local Variable</h2>";
$num = 60; /* $num outside function local_var() is a
              completely different Variable than that of
              inside local_var() */

function local_var()
{
    $num = 50; /* This $num is local to this function
                  the variable $num outside this function
                  is a completely different variable */
    echo "local num = $num <br>";
}

local_var();
echo "Variable num outside local_var() is $num ";
?>
```



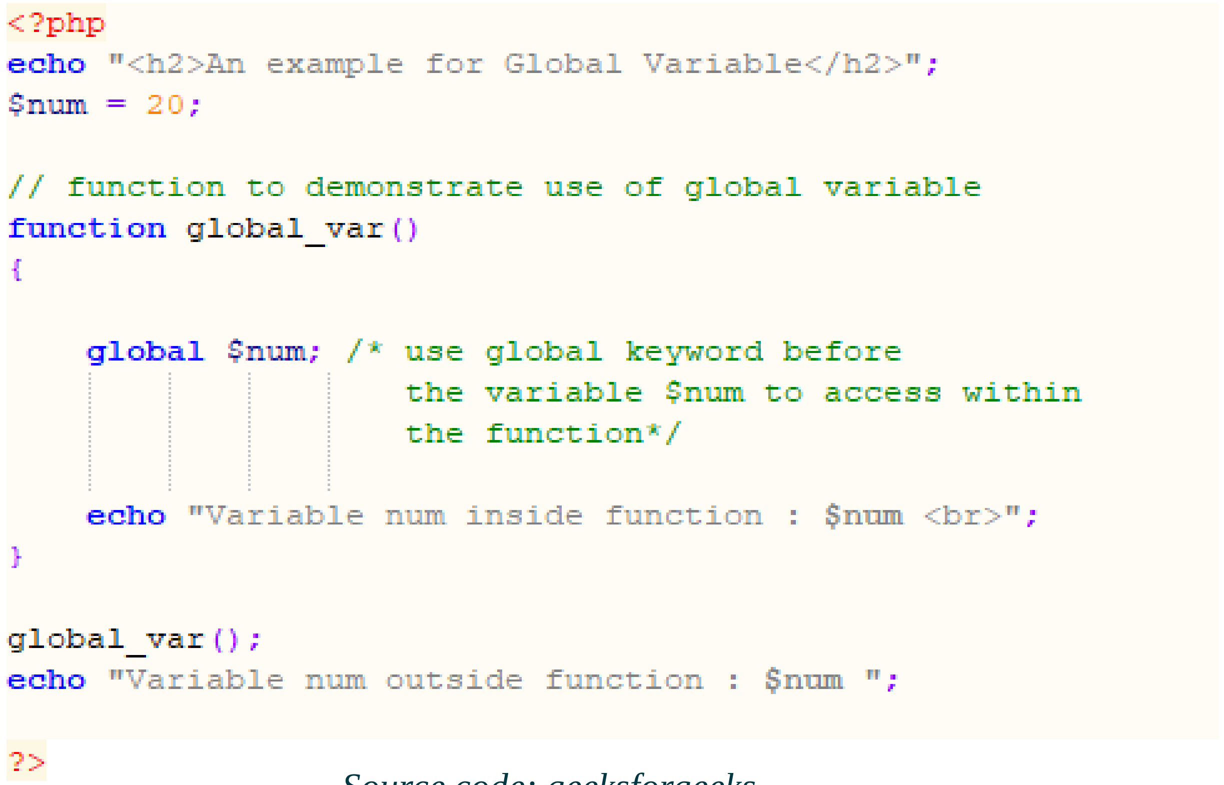
An example for Local Variable

```
local num = 50
Variable num outside local_var() is 60
```

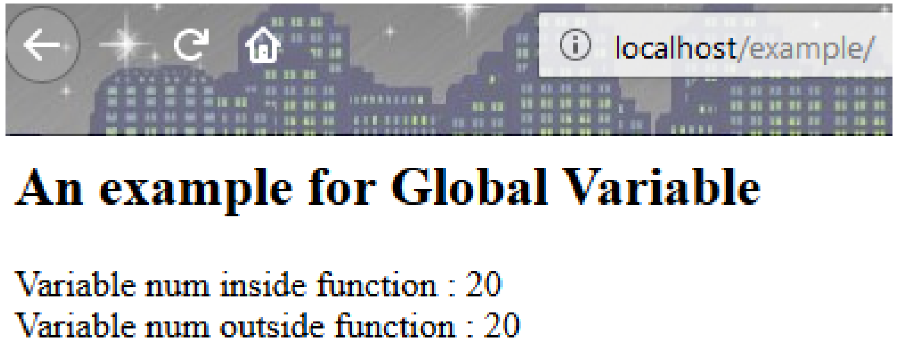
Source code: geeksforgeeks

PHP GLOBAL VARIABLE

Global variables are declared ***outside*** a function and it can only be accessed outside a function. You can access a global variable within a function using global keyword(inside the function).



```
<?php
echo "<h2>An example for Global Variable</h2>";
$num = 20;

// function to demonstrate use of global variable
function global_var()
{

    global $num; /* use global keyword before
                    the variable $num to access within
                    the function*/

    echo "Variable num inside function : $num <br>";
}

global_var();
echo "Variable num outside function : $num ";

?>
```



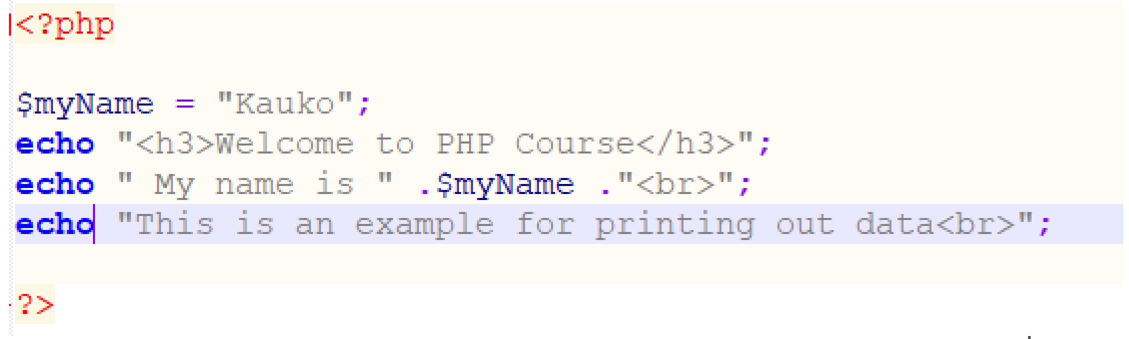
An example for Global Variable

```
Variable num inside function : 20
Variable num outside function : 20
```

Source code: geeksforgeeks

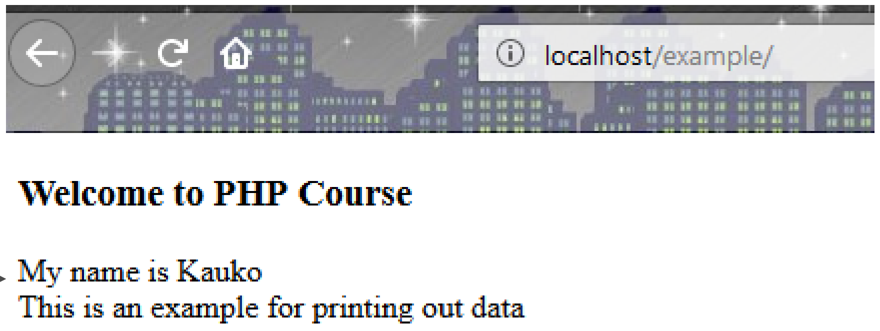
BASIC SYNTAX: echo

We use the command echo to output data to the screen, it can be used with or without parentheses: **echo or echo()**
We can include any HTML tags in PHP code which proving how easy it is to use HTML and PHP side-by-side



```php
<?php

$myName = "Kauko";
echo "<h3>Welcome to PHP Course</h3>";
echo " My name is " .$myName ."<br>";
echo "This is an example for printing out data<br>";

?>
```

Output



```
Welcome to PHP Course

My name is Kauko
This is an example for printing out data
```

Remember to run in the server!!!

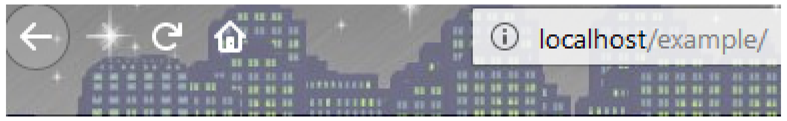
BASIC SYNTAX: print

We use the command echo to output data to the screen, it can be used with or without parentheses: **print or print()**
We can include any HTML tags in PHP code which proving how easy it is to use HTML and PHP side-by-side

```php
<?php

$myName = "Kauko";
print "<h3>Welcome to PHP Course</h3>";
print " I am " .$myName ."<br>";
print "This is an example for printing out data<br>";

?>
```

Remember to run in the server!!!

Output



```
Welcome to PHP Course

I am Kauko
This is an example for printing out data
```

PHP SUPER GLOBAL VARIABLES

Several predefined variables in PHP are "superglobals", which means that they are always accessible, regardless of scope - and you can access them from any function, class or file without having to do anything special. (*w3schools)*

The list of superglobal variables available in PHP:

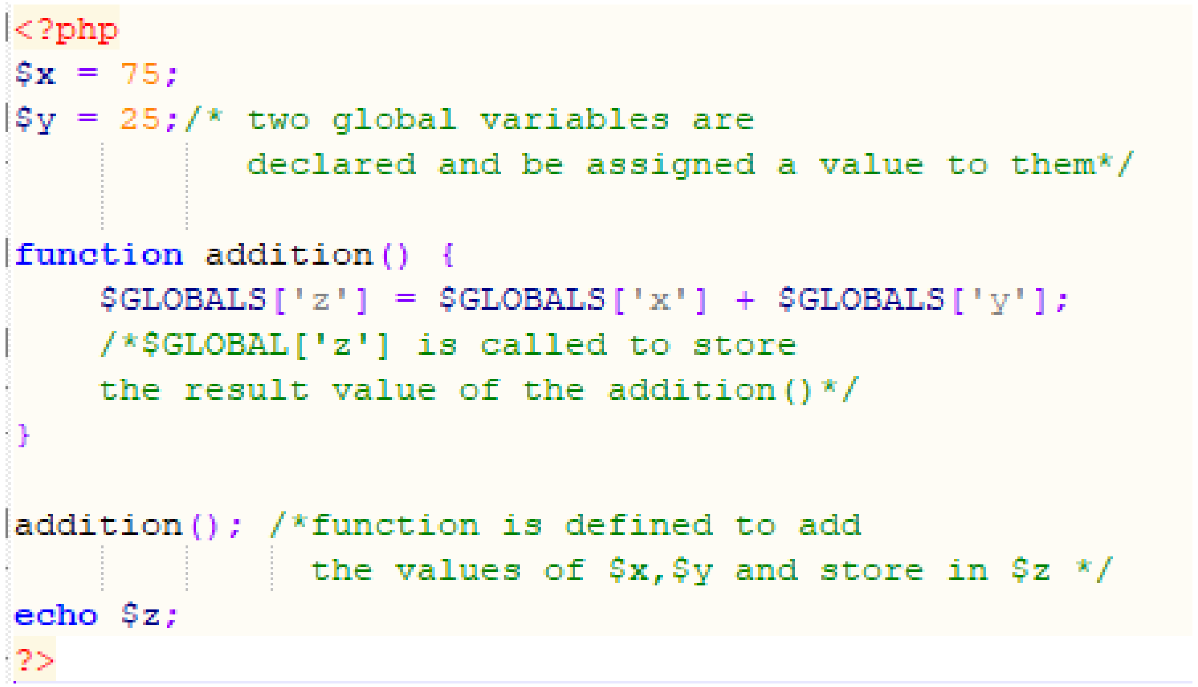
$GLOBALS
$_SERVER
$_REQUEST
$_POST
$_GET
$_FILES
$_ENV
$_COOKIE
$_SESSION

Notes: the followed example is mainly showed you how superglobal variables work. You may get confused at some point about the syntax. Nevertheless, do not worry, we will learn about them at the next few chapters.

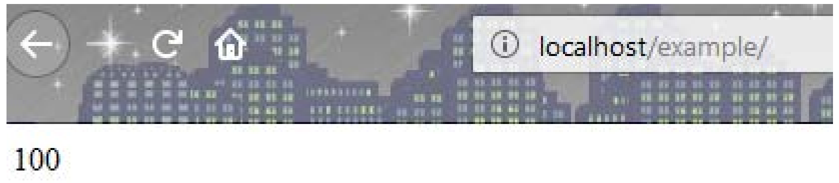
Few superglobal will not be displayed in this chapter, the rest will be explained in later chapter.

PHP SUPER GLOBAL VARIABLES: $GLOBALS

$GLOBALS is a superglobal variable which is used to access global variables from anywhere in the PHP script. PHP stores all the global variables in array $GLOBALS[index] an index that holds the global variable name, which can be accessed.



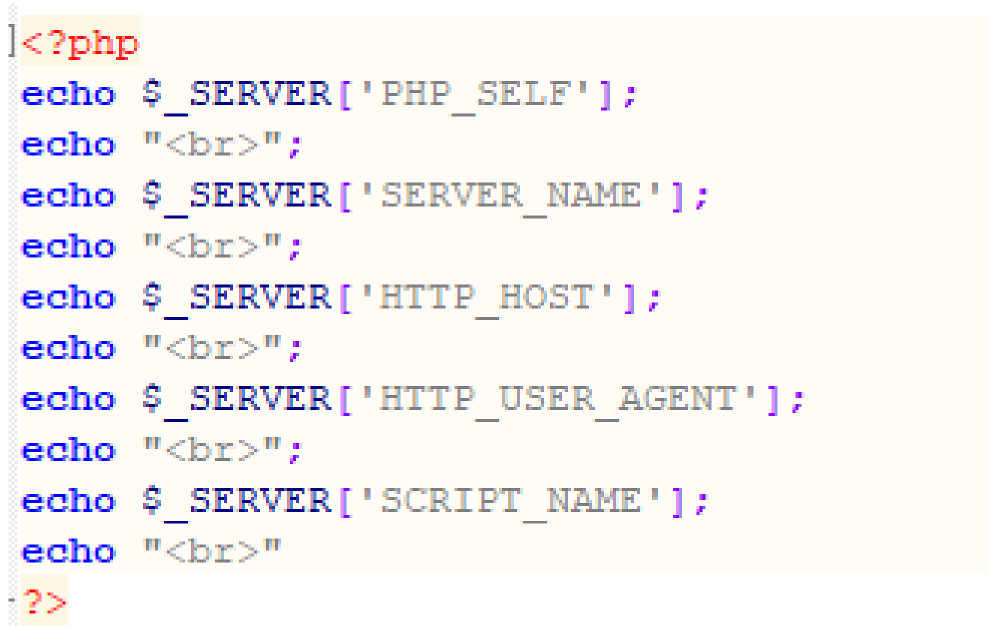
```
<?php
$x = 75;
$y = 25;/* two global variables are
           declared and be assigned a value to them*/

function addition() {
    $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
    /*$GLOBAL['z'] is called to store
    the result value of the addition()*/
}

addition(); /*function is defined to add
               the values of $x,$y and store in $z */
echo $z;
?>
```



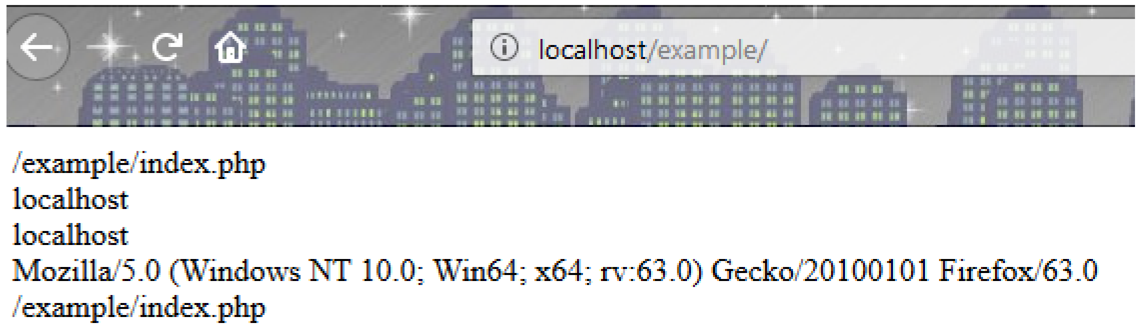
```
100
```

PHP SUPER GLOBAL VARIABLES: $_SERVER

$_SERVER : It is a PHP super global variable that stores the information about headers, paths and script locations.



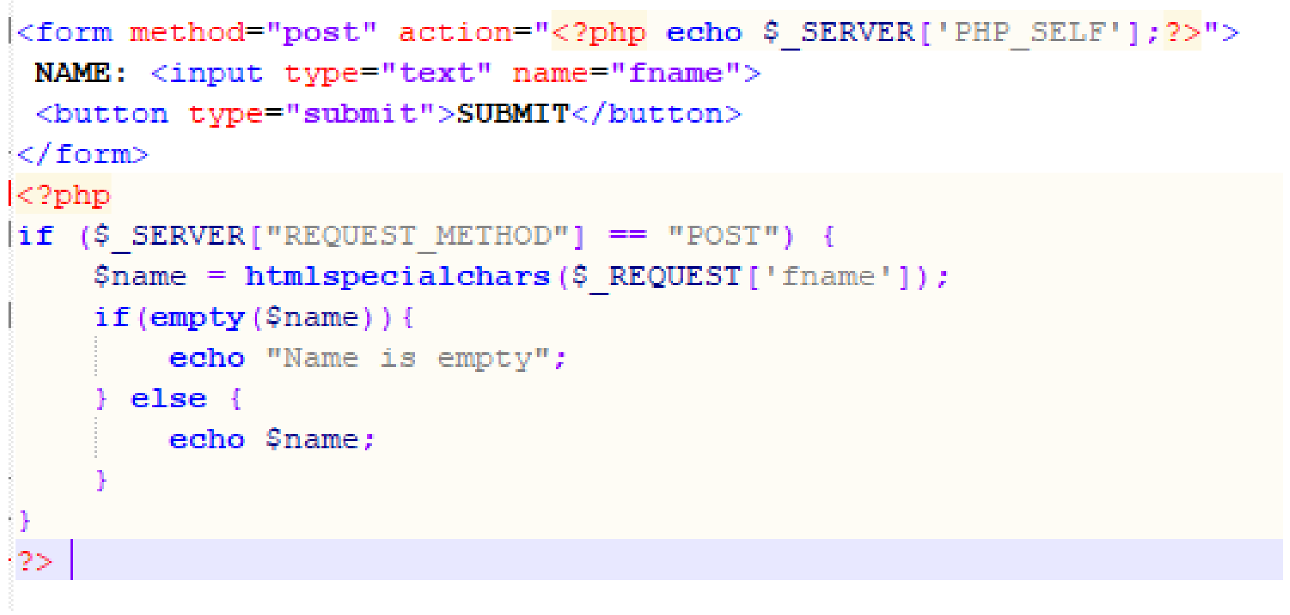
```
<?php
echo $_SERVER['PHP_SELF'];
echo "<br>";
echo $_SERVER['SERVER_NAME'];
echo "<br>";
echo $_SERVER['HTTP_HOST'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
echo "<br>"
?>
```



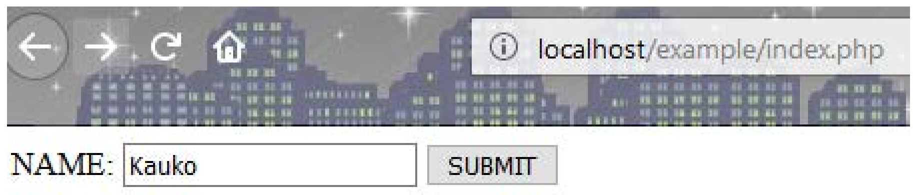
```
/example/index.php
localhost
localhost
Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0
/example/index.php
```

PHP SUPER GLOBAL VARIABLES: $_REQUEST

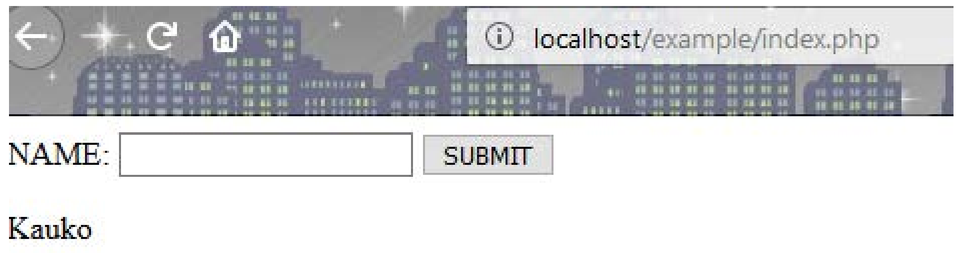
$_REQUEST : It is a superglobal variable which is used to collect the data after submitting a HTML form. $_REQUEST is not used mostly, because $_POST and $_GET perform the same task and are widely used.



```
<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
 NAME: <input type="text" name="fname">
 <button type="submit">SUBMIT</button>
</form>
<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $name = htmlspecialchars($_REQUEST['fname']);
    if(empty($name)){
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>
```

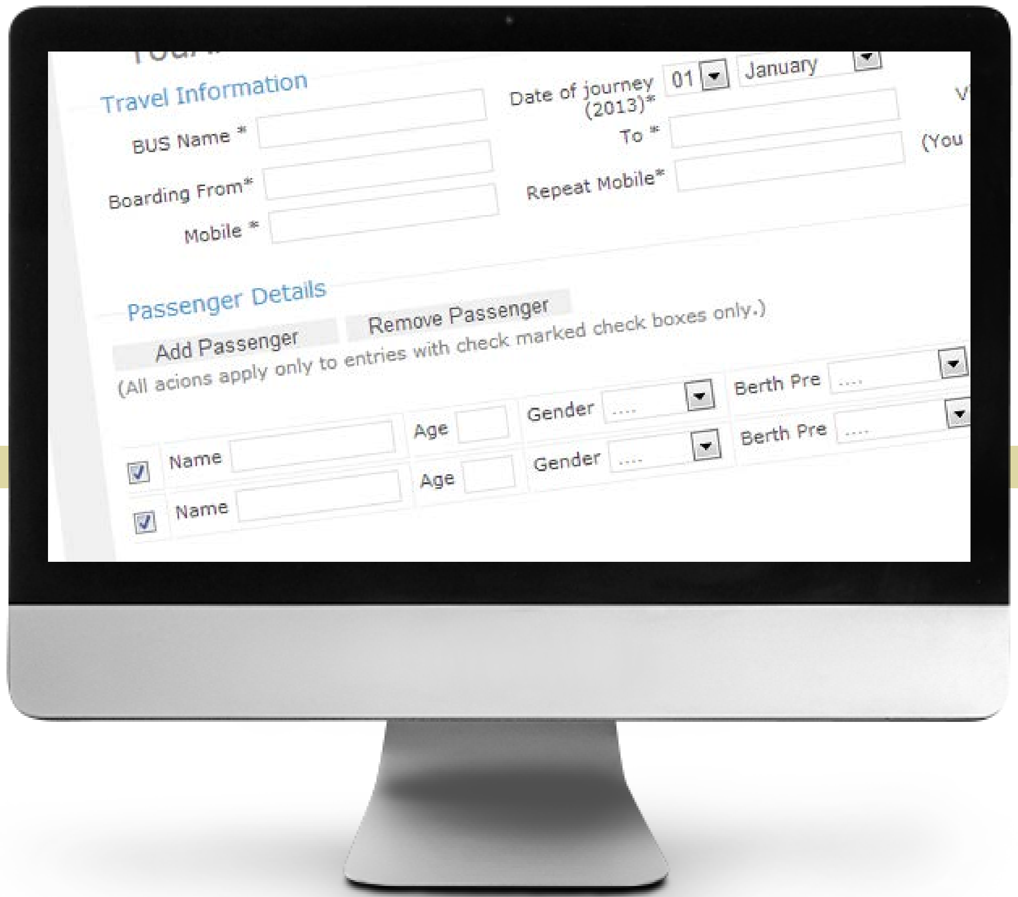


NAME: Kauko SUBMIT



NAME: SUBMIT

Kauko



Travel Information

BUS Name *

Boarding From*

Mobile *

Date of journey (2013)* 01 January

To *

Repeat Mobile*

(You

Passenger Details

Add Passenger Remove Passenger

(All acions apply only to entries with check marked check boxes only.)

☑	Name	Age	Gender	Berth Pre ....
☑	Name	Age	Gender	Berth Pre ....

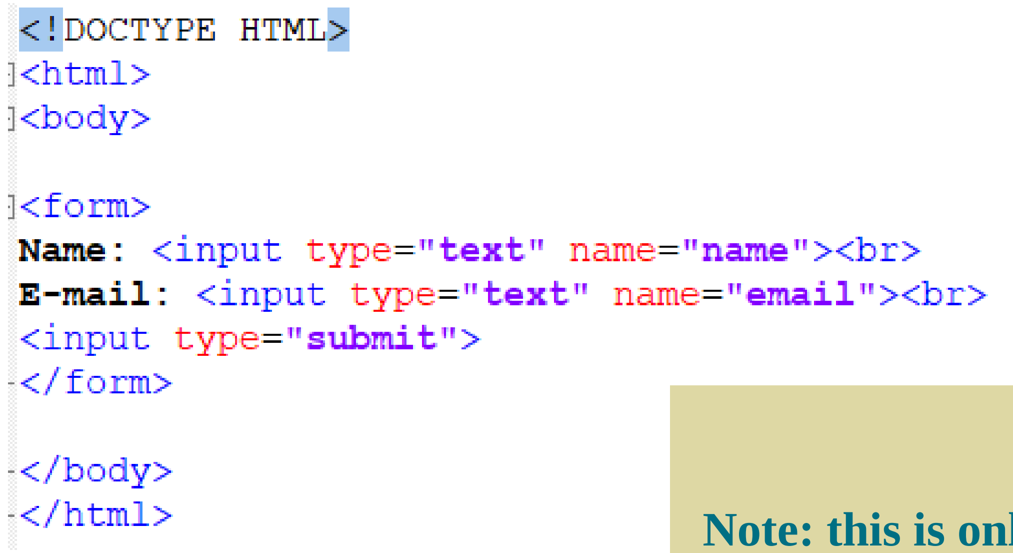
PHP FORMS

One of the most powerful features of PHP is the way it handles HTML forms

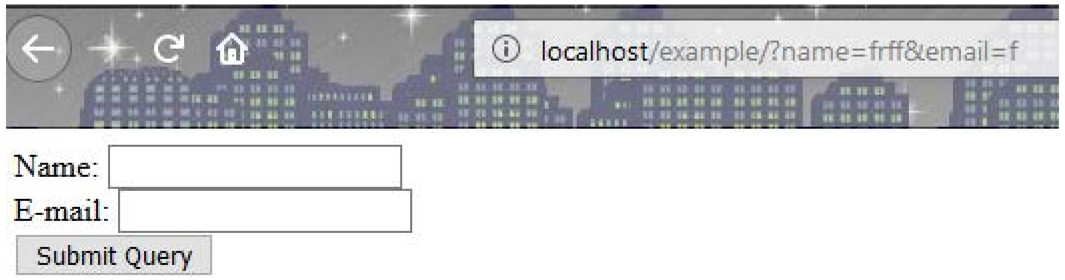
The basic concept that is important to understand is that any form element will automatically be available to your PHP scripts. Form can either be written in HTML only or in PHP.

PHP FORMS: The <form> tag

The form element creates a form for user input.



```
<!DOCTYPE HTML>
<html>
<body>

<form>
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

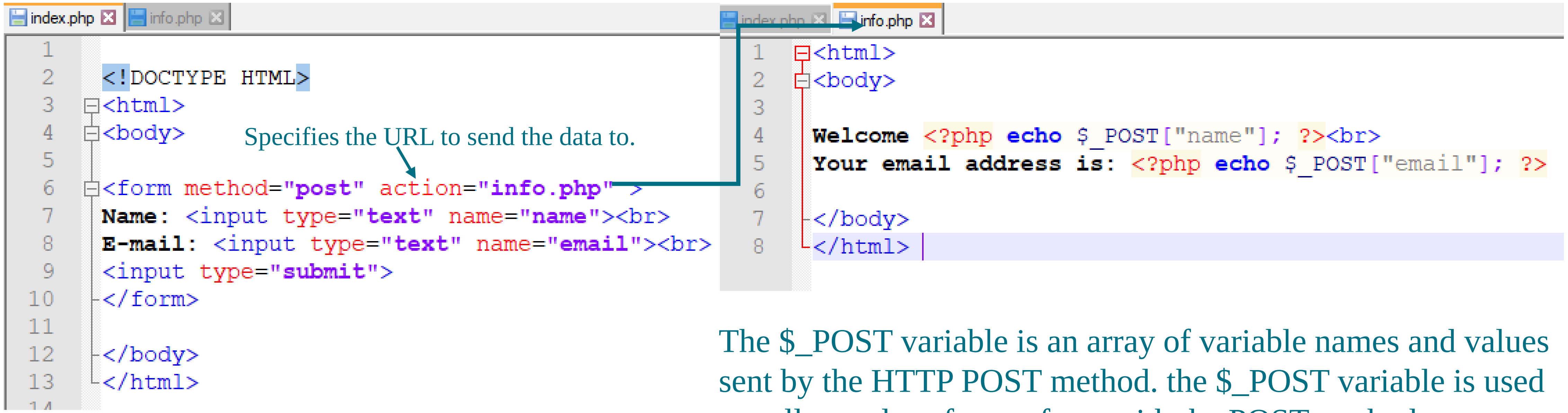


```
localhost/example/?name=frff&email=f
Name: [          ]
E-mail: [          ]
[Submit Query]
```

Note: this is only for displaying the form, the button Submit have not worked yet

PHP FORMS: POST method

When a form is submitted, all HTML variables are sent to a PHP script, for example we named it "info.php". The form data is sent with POST method. It is used to store, retrieve and send data.



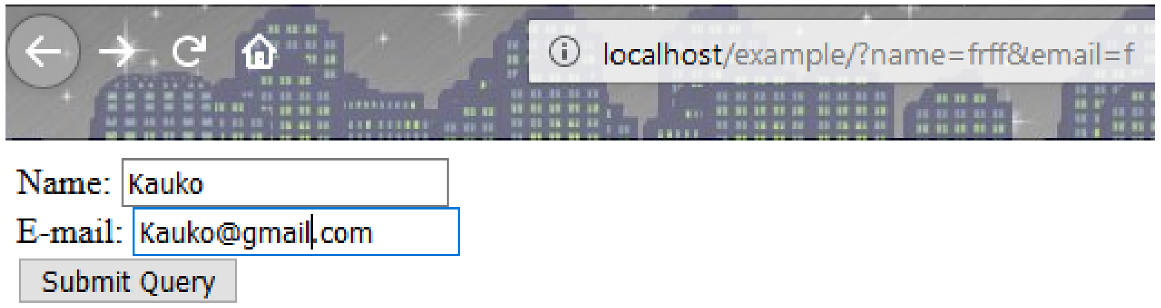
```
<!DOCTYPE HTML>
<html>
<body>

<form method="post" action="info.php" >
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

```
<html>
<body>

Welcome <?php echo $_POST["name"]; ?><br>
Your email address is: <?php echo $_POST["email"]; ?>

</body>
</html>
```

The $_POST variable is an array of variable names and values sent by the HTTP POST method. the $_POST variable is used to collect values from a form with the POST method.

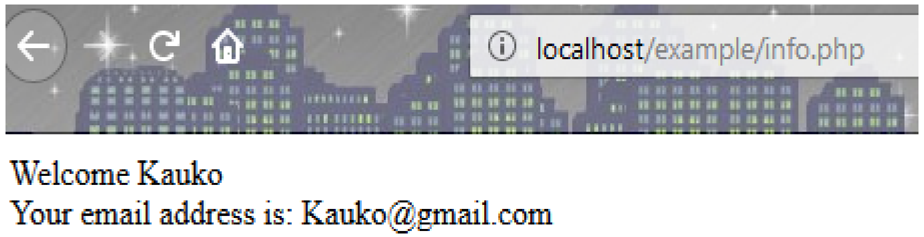
PHP FORMS: POST method

Output would look like this:



localhost/example/?name=frff&email=f

Name: Kauko
E-mail: Kauko@gmail.com
Submit Query

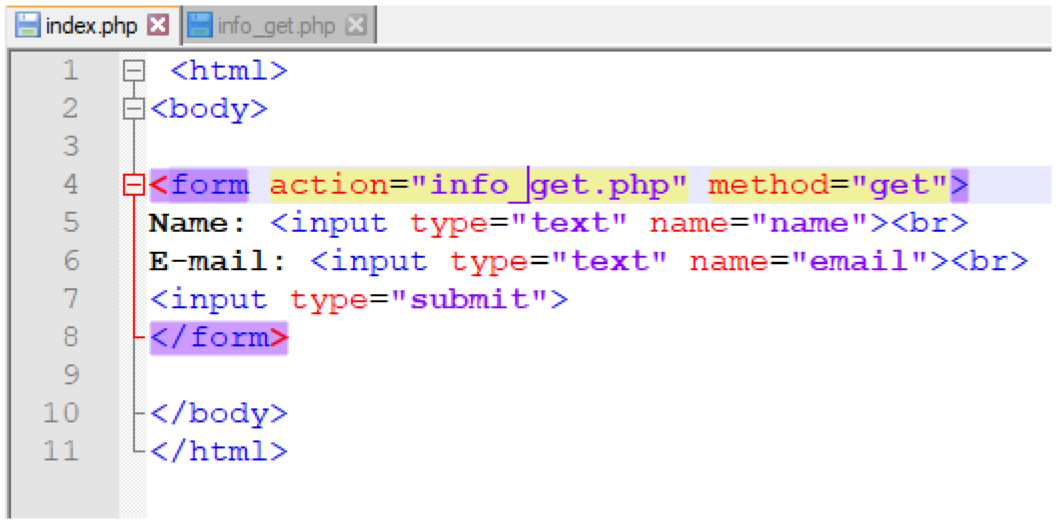


localhost/example/info.php

```
Welcome Kauko
Your email address is: Kauko@gmail.com
```

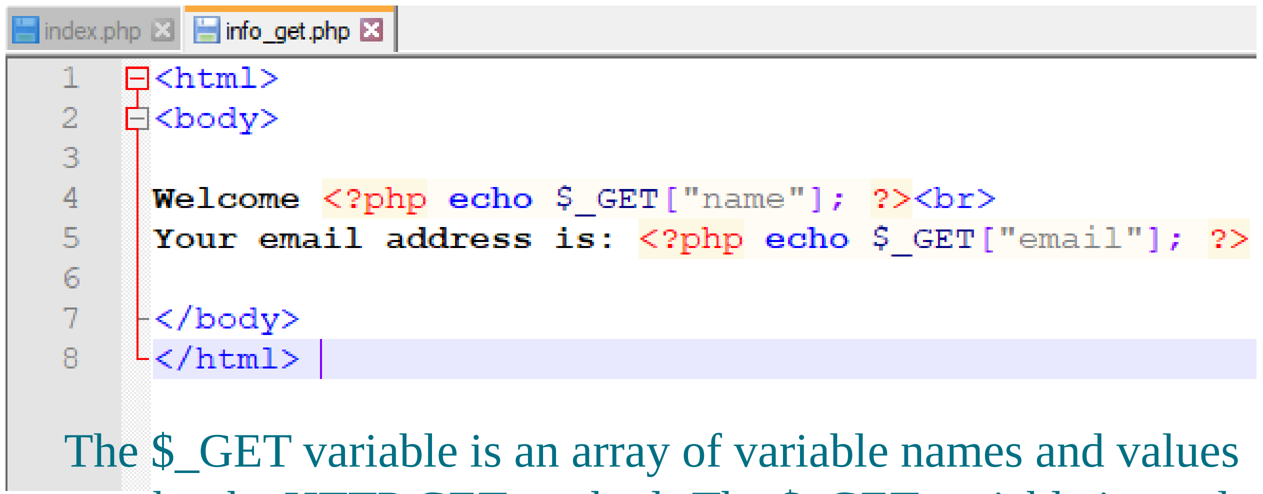
PHP FORMS: GET method

It similar to the POST method. Therefore, they have a few differences. GET method (default) is used to retrieve data.



index.php | info_get.php

```
 <html>
<body>

<form action="info_get.php" method="get">
Name: <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit">
</form>

</body>
</html>
```

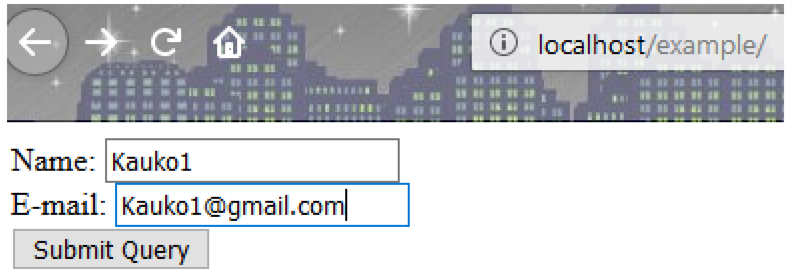


index.php | info_get.php

```
<html>
<body>

Welcome <?php echo $_GET["name"]; ?><br>
Your email address is: <?php echo $_GET["email"]; ?>

</body>
</html>
```

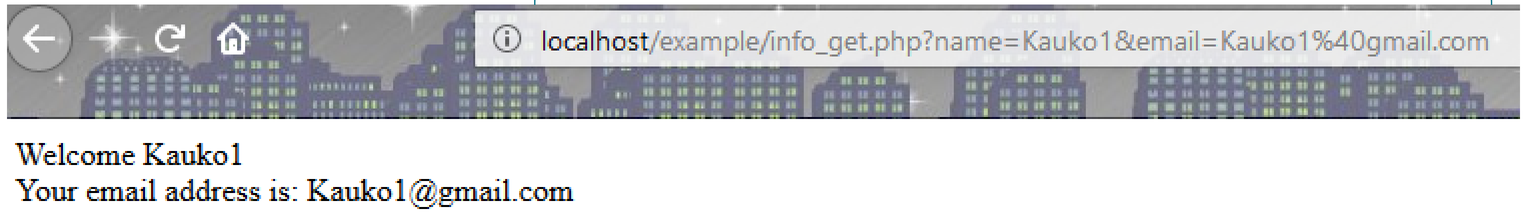
The $_GET variable is an array of variable names and values sent by the HTTP GET method. The $_GET variable is used to collect values from a form with GET method.

PHP FORMS: GET method


localhost/example/

Name: Kauko1
E-mail: Kauko1@gmail.com

Now we can see clearly the different between GET and POST. unlike using POST method, the information sent is visible to visitors in the URL(you can go back to POST slide to see the difference at the URL bar)


localhost/example/info_get.php?name=Kauko1&email=Kauko1%40gmail.com

Welcome Kauko1
Your email address is: Kauko1@gmail.com

WHEN TO USE POST AND GET?

GET	POST
Information sent from a form with the GET method is visible to everyone (all variable names and values are displayed in the URL). GET also has limits on the amount of information to send. The limitation is about 2000 characters. However, because the variables are displayed in the URL, it is possible to bookmark the page. This can be useful in some cases. GET may be used for sending non-sensitive data. **Note: GET should NEVER be used for sending passwords or other sensitive information!**	Information sent from a form with the POST method is invisible to others (all names/values are embedded within the body of the HTTP request) and has no limits on the amount of information to send. Moreover POST supports advanced functionality such as support for multi-part binary input while uploading files to server. However, because the variables are not displayed in the URL, it is not possible to bookmark the page. **Developers prefer POST for sending form data.**

Source: w3schools

PHP | LOOPS

A loop allows executing a statement or block of statements repeatedly, unless a specific condition is met.

Types of loop

1

for loop

2

while loop

3

do-while loop

4

foreach loop

PHP|LOOPS: for loop

for loop is used when the we know how many times the script needs to execute, as long as a given condition has the boolean value TRUE

Initialize the loop counter value

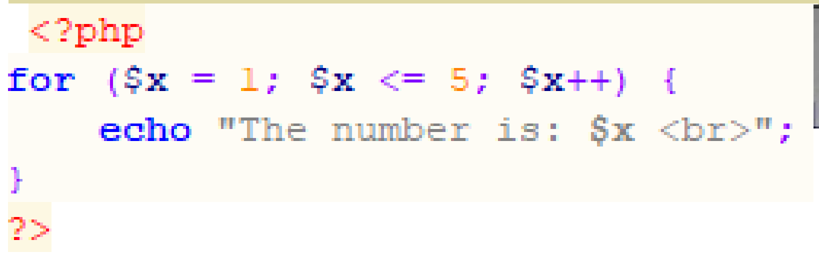
this expression increments/decrements the loop variable by some value

for(initialization_expression; loop_condition; increment_expression){

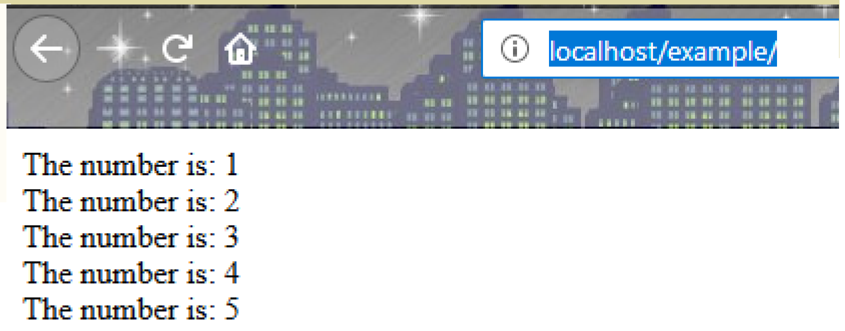
// statements

}

Evaluated for each loop iteration. If it evaluates true then we will execute and go to update expression,otherwise the loop ends.



```
<?php
for ($x = 1; $x <= 5; $x++) {
    echo "The number is: $x <br>";
}
?>
```



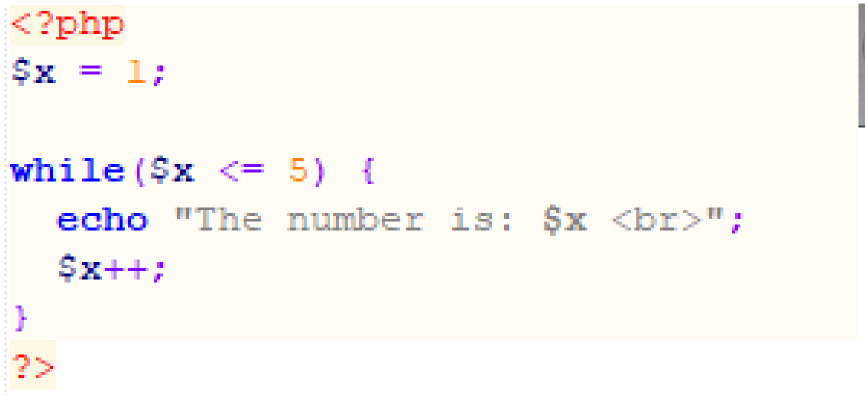
```
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
```

PHP|LOOPS: while loop

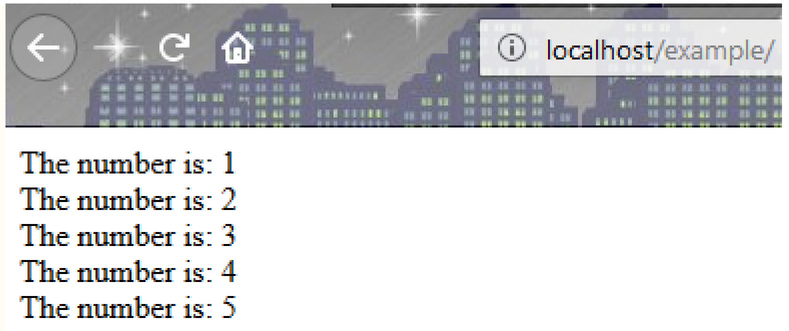
while loop is used when we execute a block of code as long as the specified condition is true.

```
while (condition is true) {

    code to be executed;

}
```

it first checks the condition at the start of the loop and if its true then it enters the loop and executes the block of statements, and goes on executing it as long as the condition holds true.



```
<?php
$x = 1;

while($x <= 5) {
   echo "The number is: $x <br>";
   $x++;
}
?>
```

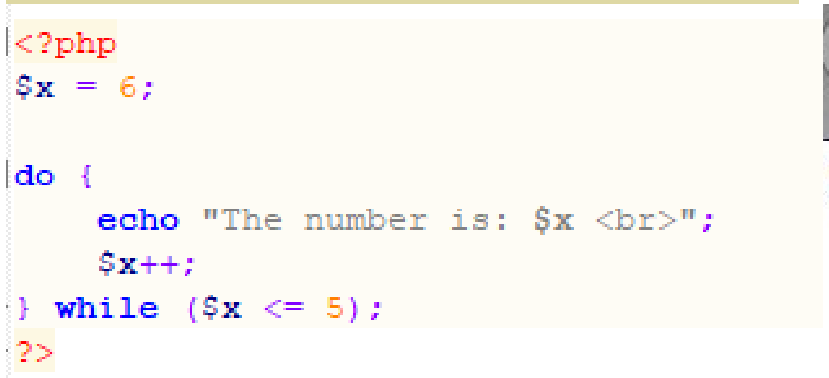


```
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
```

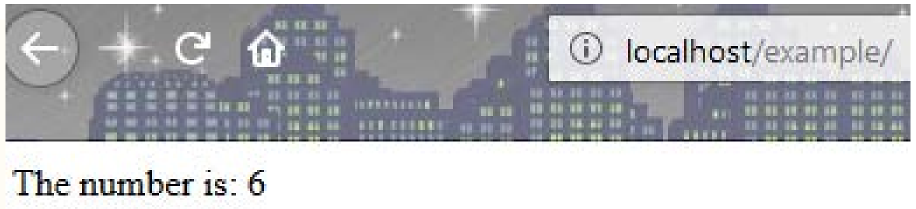
PHP|LOOPS: do-while loop

do-while loop executes the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.
It is tested AFTER executing the statements within the loop.

```
do {
    code to be executed;
} while (condition is true);
```



```
<?php
$x = 6;

do {
    echo "The number is: $x <br>";
    $x++;
} while ($x <= 5);
?>
```

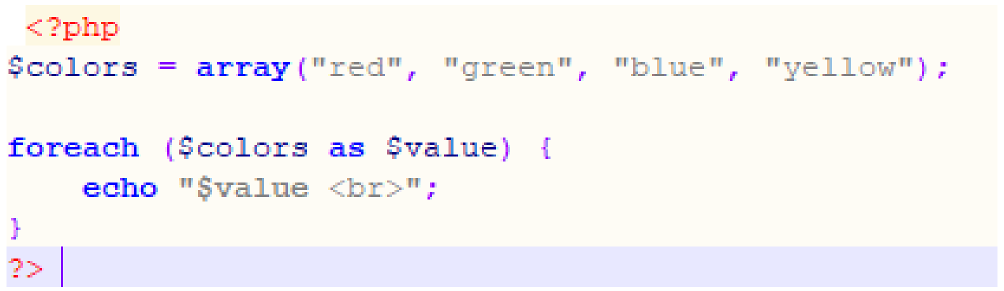


```
The number is: 6
```

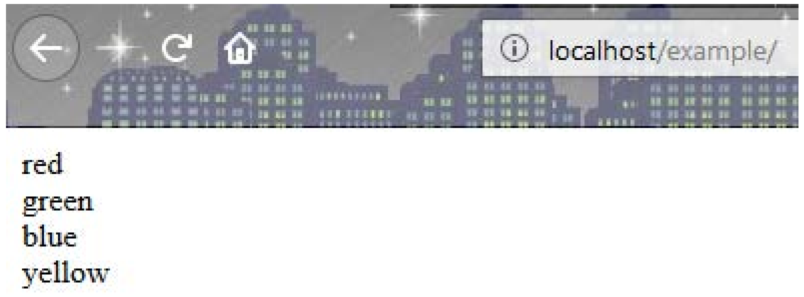
PHP | LOOPS: foreach loop

foreach loop works only on arrays, and is used to loop through each key/value pair in an array.

```
foreach ($array as $value) {
    code to be executed;
}
```



```
<?php
$colors = array("red", "green", "blue", "yellow");

foreach ($colors as $value) {
    echo "$value <br>";
}
?>
```



```
red
green
blue
yellow
```