

XAMPP Server

XAMPP Server

- XAMPP stands for Cross-Platform (X), Apache (A), MySQL (M), **PHP** (P) and **Perl**(P).
- It is a simple, light-weighted Apache server that makes it extremely easy for developers to create a local http server
- Free software

Features and Requirements of XAMPP

- Requires only one .exe or .zip file for configurations (no configuration of various components of server is required)
- It regularly updates latest release of Apache/MySQL/PHP/Perl
- Installing XAMPP takes less time than installing individual components
- Self contained and multiple instances of XAMPP can exist on same computer

Starting XAMPP Server

- Open XAMPP control panel
- Start Apache, MySQL services(required services)
- Minimize the Control Panel
- **Save your folder(containing multiple webpages) in htdocs folder inside xampp folder** (.html/.php/.js/.css)
- Check if the XAMPP server has started properly by
 - Create a web page
 - Go to Web browser (type : **localhost/sample**)
 - View the page

Basics of PHP

What is PHP?

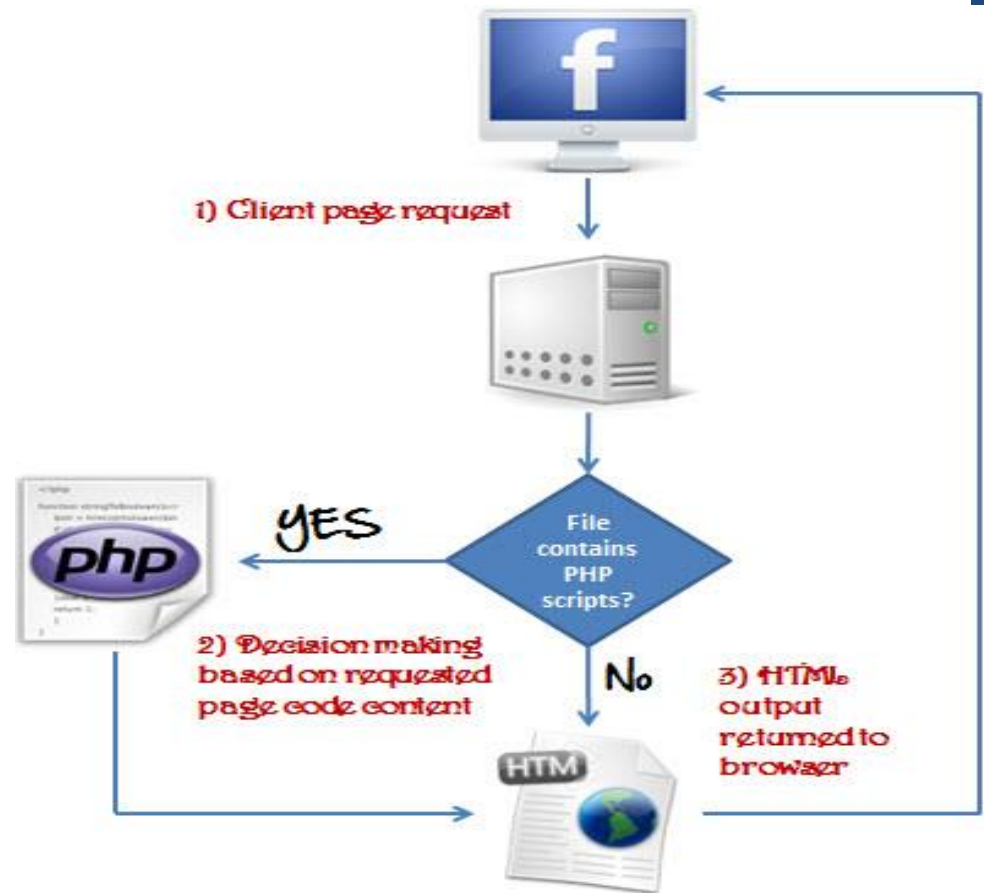
- PHP is an acronym for "PHP: Hypertext Preprocessor"
- PHP is a widely-used, open source scripting language
- PHP scripts are executed on the server

What is a PHP File?

- PHP files can contain text, HTML, CSS, JavaScript, and PHP code

- PHP files have extension

".php"



What Can PHP Do?

- Generate dynamic page content
- Create, open, read, write, delete, and close files on the server
- Collect form data
- Send and receive cookies
- Add, delete, modify data in your database
- To control user-access
- Data encryption

With PHP can generate output HTML, images, PDF files, and even Flash movies.

Why PHP?

- PHP runs on various platforms (Windows, Linux, Unix, Mac OS X, etc.)
- PHP is compatible with almost all servers used today (Apache, IIS etc.)
- PHP supports a wide range of databases
- PHP is free. Download it from the official PHP resource: www.php.net
- PHP is easy to learn and runs efficiently on the server side

Install PHP

- To install PHP, suggest you to install AMP (Apache, MySQL, PHP) software stack.
- It is available for all operating systems. There are many AMP options available in the market that are given below:
- **WAMP** for Windows
(<http://www.wampserver.com/en/>)
- **LAMP** for Linux
(<http://csg.sph.umich.edu/abecasis/LAMP/download/>)
- **MAMP** for Mac
(<https://www.mamp.info/en/downloads/>)
- **XAMPP** (Cross, Apache, MySQL, PHP, Perl) for Cross Platform
(<https://www.apachefriends.org/download.html>)

Example of PHP

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1>My first PHP page</h1>
```

```
<?php
```

```
echo "Hello World!";
```

```
?>
```

```
</body>
```

```
</html>
```

In PHP, all keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are NOT case-sensitive. However; all variable names are case-sensitive.

PHP echo

- PHP echo is a language construct not a function, so you don't need to use parenthesis with it.
- But if you want to use more than one parameters, it is required to use parenthesis.
- **Syntax:**

```
void echo ( string $arg1 [, string $... ] )
```

- PHP echo statement can be used to print string, multi line strings, escaping characters, variable, array etc.

Example: PHP echo

<?php

echo "Hello by PHP echo"; *//print string*

echo "Hello by PHP echo
this is multi line
text printed by
PHP echo statement
";

// multi line string

echo "Hello escape \"sequence\" characters"; *//escape characters*

\$msg="Hello JavaTpoint PHP";
echo "Message is: \$msg";

//print variable value

?>

PHP print

- Like PHP echo, PHP print is a language construct, so you don't need to use parenthesis with the argument list. Unlike echo, it always returns 1.

- **Syntax:**

`int print(string $arg)`

- PHP print statement can be used to print string, multi line strings, escaping characters, variable, array etc.

Example: PHP print

<?php

print "Hello by PHP echo"; *//print string*

print "Hello by PHP echo
this is multi line
text printed by
PHP echo statement
";

// multi line string

print "Hello escape \"sequence\" characters"; *//escape characters*

\$msg="Hello PHP";

print "Message is: \$msg"; *//print variable value*

?>

PHP echo and print Statements

- echo and print are more or less the same.
- Used to output data to the screen.
- **Differences:**
 - echo has no return value while print has a return value of 1 so it **can be used in expressions**
 - echo can take multiple parameters (although such usage is rare) while print can take one argument.
 - echo is marginally faster than print.

PHP Variables

- A variable starts with the \$ sign, followed by the name of the variable
- A variable name must start with a letter or the underscore character
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- A variable name **cannot start with a number**
- Variable names **are case-sensitive** (\$age and \$AGE are two different variables)

PHP has three different variable scopes:

- local
- global
- static

Example: Variables

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$txt = "Hello world!";
```

```
$x = 5;
```

```
$y = 10.5;
```

```
echo $txt;
```

```
echo "<br>";
```

```
echo $x;
```

```
echo "<br>";
```

```
echo $y;
```

```
?>
```

```
</body> </html>
```

Example: Variables

```
<?php  
$txt = "K J Somaiya College of Engineering";  
echo "Name of my College is $txt";  
?>
```

Same as

```
<?php  
$txt = "K J Somaiya College of Engineering";  
echo "Name of my College is " . $txt . "!!";  
?>
```

What will be the output of

```
<?php  
$x =15;  
$y =41;  
echo $x + $y;  
?>
```

Output:

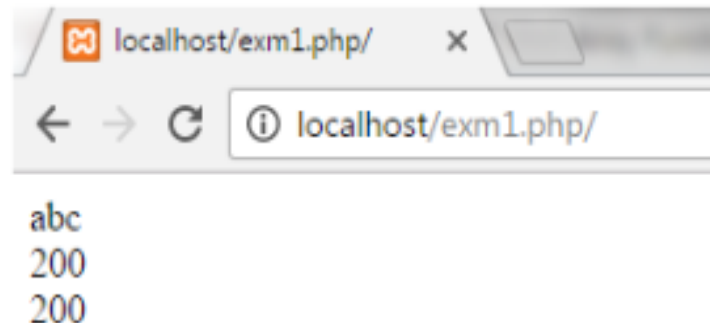
1541 Or 56

PHP \$ and \$\$ Variables

- The **\$var** (single dollar) is a normal variable with the name var that stores any value like string, integer, float, etc.
- The **\$\$var** (double dollar) is a reference variable that stores the value of the \$variable inside it.

```
<?php
$x = "abc";
$$x = 200;
echo $x."<br/>";
echo $$x."<br/>";
echo $abc;
?>
```

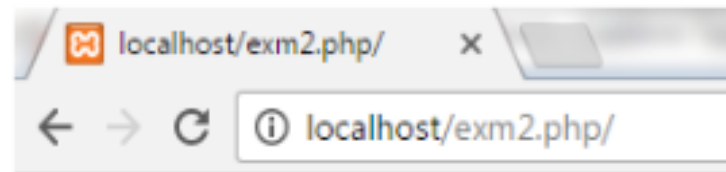
Output:



Example:

```
<?php
  $x="U.P";
  $$x="Lucknow";
  echo $x. "<br>";
  echo $$x. "<br>";
  echo "Capital of $x is " . $$x;
?>
```

Output:



U.P
Lucknow
Capital of U.P is Lucknow

PHP Constants

- A constant is an identifier (name) for a simple value.
- The value cannot be changed during the script.
- A valid constant name starts with a letter or underscore (**no \$ sign before the constant name**).
- PHP constants can be defined by 2 ways:

- **Using define() function**

Syntax:

`define(name, value, case-insensitive)`

- **Using const keyword**

- It is a language construct not a function.
- It is bit faster than define().
- It is always case sensitive.

Example: Constants

- `<?php`
define("College", "K J Somaiya College of Engineering ");
echo College;

define("College", "K J Somaiya College of Engineering ", true);
echo College;

const MESSAGE=" K J Somaiya College of Engineering ";
echo MESSAGE;

`?>`

PHP Operators

- Operators are used to perform operations on variables and values.
 - Arithmetic operators
 - Assignment operators
 - Comparison operators
 - Increment/Decrement operators
 - Logical operators
 - String operators
 - Array operators

Arithmetic Operators

Operator	Name	Example	Result
+	Addition	$\$x + \y	Sum of $\$x$ and $\$y$
-	Subtraction	$\$x - \y	Difference of $\$x$ and $\$y$
*	Multiplication	$\$x * \y	Product of $\$x$ and $\$y$
/	Division	$\$x / \y	Quotient of $\$x$ and $\$y$
%	Modulus	$\$x \% \y	Remainder of $\$x$ divided by $\$y$
**	Exponentiation	$\$x ** \y	Result of raising $\$x$ to the $\$y$ 'th power (Introduced in RUP E 6)

Arithmetic Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 10;
```

```
$y = 6;
```

```
echo $x - $y;
```

```
?>
```

```
</body>
```

```
</html>
```

Assignment Operators

Assignment	Same as...	Description
$x = y$	$x = y$	The left operand gets set to the value of the expression on the right
$x += y$	$x = x + y$	Addition
$x -= y$	$x = x - y$	Subtraction
$x *= y$	$x = x * y$	Multiplication
$x /= y$	$x = x / y$	Division
$x \%= y$	$x = x \% y$	Modulus

Assignment Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 20;
```

```
$x += 100;
```

```
echo $x;
```

```
?>
```

```
</body>
```

```
</html>
```

Comparison Operators

Operator	Name	Example	Result
==	Equal	$\$x == \y	Returns true if $\$x$ is equal to $\$y$
===	Identical	$\$x === \y	Returns true if $\$x$ is equal to $\$y$, and they are of the same type
!=	Not equal	$\$x != \y	Returns true if $\$x$ is not equal to $\$y$
<>	Not equal	$\$x <> \y	Returns true if $\$x$ is not equal to $\$y$
!==	Not identical	$\$x !== \y	Returns true if $\$x$ is not equal to $\$y$, or they are not of the same type
>	Greater than	$\$x > \y	Returns true if $\$x$ is greater than $\$y$
<	Less than	$\$x < \y	Returns true if $\$x$ is less than $\$y$

Comparison Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 100;
```

```
$y = "100";
```

```
var_dump($x === $y);
```

```
?>
```

```
</body> </html>
```

Increment / Decrement Operator

Operator	Name	Description
<code>++\$x</code>	Pre-increment	Increments \$x by one, then returns \$x
<code>\$x++</code>	Post-increment	Returns \$x, then increments \$x by one
<code>--\$x</code>	Pre-decrement	Decrements \$x by one, then returns \$x
<code>\$x--</code>	Post-decrement	Returns \$x, then decrements \$x by one

Increment / Decrement Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 10;
```

```
echo ++$x;
```

```
?>
```

```
</body>
```

```
</html>
```

Logical Operator

Operator	Name	Example	Result
and	And	\$x and \$y	True if both \$x and \$y are true
or	Or	\$x or \$y	True if either \$x or \$y is true
xor	Xor	\$x xor \$y	True if either \$x or \$y is true, but not both
&&	And	\$x && \$y	True if both \$x and \$y are true
	Or	\$x \$y	True if either \$x or \$y is true
!	Not	!\$x	True if \$x is not true

Logical Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 100;
```

```
$y = 50;
```

```
if ($x == 100 && $y == 50) {
```

```
    echo "Hello world!";
```

```
}
```

```
?>
```

```
</body>
```

```
</html>
```

Comments in PHP

```
<html>
```

```
<body>
```

```
<?php
```

```
// This is a single-line comment
```

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

```
/*
```

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

```
*/
```

```
// You can also use comments to leave out parts of a code line
```

```
$x = 5 /* + 15 */ + 5;
```

```
echo $x;
```

```
?> </body> </html>
```

PHP Data Types

- String
- Integer
- Float (also called double)
- Boolean
- Array
- Object
- NULL
- Resource

Scalar Data types

Compound Data Types

Special Data types

The PHP `var_dump()` function returns the data type and value

String, Integer, Float and Boolean: Example

```
<?php
```

```
$x = "Hello world!";
```

```
$y = 'Hello world!';
```

```
echo $x;
```

```
echo "<br>";
```

```
echo $y;
```

```
$x = 5985;
```

```
var_dump($x);
```

```
$x = 10.365;
```

```
var_dump($x);
```

```
$m=true;
```

```
echo $m;
```

```
?>
```

Array: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$animals = array("Elephant ","Dog","Tiger");
```

```
var_dump($animals);
```

```
?>
```

```
</body>
```

```
</html>
```

Output:

```
array(3) { [0]=> string(8) "Elephant" [1]=> string(3) "Dog" [2]=> string(5) "Tiger" }
```

Object: Example

- An object is a data type which stores data and information on how to process that data.

```
<?php
class Car {
    function Car() {
        $this->model = "VW";
    }
}
// create an object
$herbie = new Car();

// show object properties
echo $herbie->model;
?>
```

Output:
VW

NULL :Example

- Null is a special data type which can have only one value: NULL.
- A variable of data type NULL is a variable that has no value assigned to it.
- **Tip: If a variable is created without a value, it is automatically assigned a value of NULL.**

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

```
<?php  
$x = "Hello world!";  
$x = null;  
var_dump($x);  
?>
```

```
</body>  
</html>
```

Output:
null

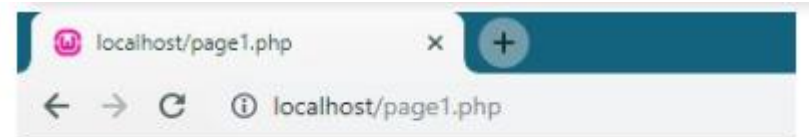
Resource: Example

- It refers the external resources like database connection, FTP connection, file pointers, etc.
- In simple terms, a resource is a special variable which carrying a reference to an external resource.
- Example:

```
<?php
```

```
$conn = ftp_connect("127.0.0.1") or die("Could not connect");  
echo get_resource_type($conn);
```

```
?>
```



FTP Buffer

Conditional Statement

- **if statement** - executes some code if one condition is true
- **if...else statement** - executes some code if a condition is true and another code if that condition is false
- **if...elseif....else statement** - executes different codes for more than two conditions
- **switch statement** - selects one of many blocks of code to be executed

if statement

- The if statement executes some code if one condition is true.

- **Example:**

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

```
<?php
$t = date("H");
```

```
if ($t < "20") {
    echo "Have a good day!";
}
?>
```

```
</body>
</html>
```

if...else Statement

- The if....else statement executes some code if a condition is true and another code if that condition is false.

- Example:

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

```
<?php
$t = date("H");
```

```
if ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

```
</body>
</html>
```

if...elseif...else Statement

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

```
<?php
$t = date("H");
echo "<p>The hour (of the server) is " . $t;
echo ", and will give the following message:</p>";
```

```
if ($t < "10") {
    echo "Have a good morning!";
} elseif ($t < "20") {
    echo "Have a good day!";
} else {
    echo "Have a good night!";
}
?>
```

```
</body>
</html>
```

switch Statement

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$favcolor = "red";
```

```
switch ($favcolor) {
```

```
    case "red":
```

```
        echo "Your favorite color is red!";
```

```
        break;
```

```
    case "blue":
```

```
        echo "Your favorite color is blue!";
```

```
        break;
```

```
    case "green":
```

```
        echo "Your favorite color is green!";
```

```
        break;
```

```
    default:
```

```
        echo "Your favorite color is neither red, blue, nor green!";
```

```
}
```

```
?>
```

```
</body> </html>
```

PHP Loops

- **while** - loops through a block of code as long as the specified condition is true
- **do...while** - loops through a block of code once, and then repeats the loop as long as the specified condition is true
- **for** - loops through a block of code a specified number of times
- **foreach** - loops through a block of code for each element in an array

while Loop

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 1;
```

```
while($x <= 5) {
```

```
    echo "The number is: $x <br>";
```

```
    $x++;
```

```
}
```

```
?>
```

```
</body>
```

```
</html>
```

do...while Loop

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = 1;
```

```
do {
```

```
    echo "The number is: $x <br>";
```

```
    $x++;
```

```
} while ($x <= 5);
```

```
?>
```

```
</body>
```

```
</html>
```

for Loop

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
for ($x = 0; $x <= 10; $x++) {  
    echo "The number is: $x <br>";  
}
```

```
?>
```

```
</body>
```

```
</html>
```

foreach loop

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

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

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

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

```
</body>
</html>
```

PHP Functions

- PHP function is a piece of code that can be reused many times. It can take input as argument list and return value. There are **thousands of built-in functions in PHP**.
- In PHP, we can define **Conditional function**, **Function within Function** and **Recursive function** also.
- Advantage of PHP Functions:
- **Code Reusability:**
 - PHP functions are defined only once and can be invoked many times, like in other programming languages.
- **Less Code:**
 - It saves a lot of code because you don't need to write the logic many times. By the use of function, you can write the logic only once and reuse it.
- **Easy to understand:**
 - PHP functions separate the programming logic. So it is easier to understand the flow of the application because every logic is divided in the form of functions.

User Defined Functions

- A function is a block of statements that can be used repeatedly in a program.
- A function will not execute immediately when a page loads.
- A function will be executed by a call to the function.

Call by value:

```
<?php
function trialFunction($Name, $Marks)
{
    echo "$Name got $Marks .<br>";
}
trialFunction("Amit","97");
trialFunction("Ajay","78");
trialFunction("Zoya","83");
?>
```

PHP Call By Reference

- Value passed to the function doesn't modify the actual value by default (call by value).

But we can do so by passing value as a reference.

- By default, value passed to the function is call by value.
- To pass value as a reference, you need to use ampersand (&) symbol before the argument name.
- **Example:**

```
<?php
function adder(&$str2)
{
    $str2 .= 'Call By Reference';
}
$str = 'Hello ';
adder($str);
echo $str;
?>
```

Output:

Hello Call By Reference

PHP Function: Default Argument Value

```
<?php
function sayHello($name="Sonoo"){
echo "Hello $name<br/>";
}
sayHello("Rajesh");
sayHello();//passing no value
sayHello("John");
?>
```

Output:

Hello Rajesh
Hello Sonoo
Hello John

PHP Function: Returning Value

```
<?php  
function cube($n)  
{  
return $n*$n*$n;  
}  
echo "Cube of 3 is: ".cube(3);  
?>
```

Output:

Cube of 3 is: 27

PHP Recursive Function

```
<?php
function display($number) {
    if($number<=5){
        echo "$number <br/>";
        display($number+1);
    }
}
```

```
display(1);
```

```
?>
```

Output:

1
2
3
4
5

PHP String Functions

- **strlen()**
returns the length of a string
- **str_word_count()**
counts the number of words in a string
- **strrev()**
reverses a string
- **strpos()**
searches for a specific text within a string (returns position of first match)
- **str_replace()**
replaces some characters with some other characters in a string.

PHP Arrays

- PHP array is an ordered map (contains value on the basis of key). It is used to hold multiple values of similar type in a single variable.
- **Advantage of PHP Array**
 - **Less Code:**

We don't need to define multiple variables.
 - **Easy to traverse:**

By the help of single loop, we can traverse all the elements of an array.
- **PHP Array Types**
 1. Indexed Array
 2. Associative Array
 3. Multidimensional Array

Example: Arrays

```
<?php
$season=array("summer","winter","spring","autumn");
echo "Season are: $season[0], $season[1], $season[2] and $season[3]";
?>
```

Output:

Season are: summer, winter, spring and autumn

```
<?php
$salary=array("Sonoo"=>"350000","John"=>"450000","Kartik"=>"200000");
echo "Sonoo salary: ".$salary["Sonoo"]."<br/>";
echo "John salary: ".$salary["John"]."<br/>";
echo "Kartik salary: ".$salary["Kartik"]."<br/>";
?>
```

Output:

Sonoo salary: 350000

John salary: 450000

Kartik salary: 200000

Example: Arrays

```
<?php
$emp = array
(
    array(1,"sonoo",400000),
    array(2,"john",500000),
    array(3,"rahul",300000)
);

for ($row = 0; $row < 3; $row++) {
    for ($col = 0; $col < 3; $col++) {
        echo $emp[$row][$col]." ";
    }
    echo "<br/>";
}
?>
```

Output:

```
1 sonoo 400000
2 john 500000
3 rahul 300000
```

Array Operators

Operator	Name	Example	Result
+	Union	$\$x + \y	Union of $\$x$ and $\$y$
==	Equality	$\$x == \y	Returns true if $\$x$ and $\$y$ have the same key/value pairs
===	Identity	$\$x === \y	Returns true if $\$x$ and $\$y$ have the same key/value pairs in the same order and of the same types
!=	Inequality	$\$x != \y	Returns true if $\$x$ is not equal to $\$y$
<>	Inequality	$\$x <> \y	Returns true if $\$x$ is not equal to $\$y$

Array Operators: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$x = array("a" => "red", "b" => "green");
```

```
$y = array("c" => "blue", "d" => "yellow");
```

```
var_dump($x != $y);
```

```
?>
```

```
</body>
```

```
</html>
```


Array Functions(1)

PHP array_change_key_case() function

- **Syntax**

array array_change_key_case (**array** \$array [, int \$case = CASE_LOWER])

- **Note:** It changes case of key only.

- **Example**

```
<?php
```

```
$salary=array("Sonoo"=>"550000","Vimal"=>"250000","Ratan"=>"200000");
```

```
print_r(array_change_key_case($salary,CASE_UPPER));
```

```
?>
```

Output:

Array ([SONOO] => 550000 [VIMAL] => 250000 [RATAN] => 200000)

Array Functions(2)

PHP array_chunk() function

- PHP array_chunk() function splits array into chunks. By using array_chunk() method, you can divide array into many parts.

- **Syntax**

array array_chunk (**array** \$array , int \$size [, bool \$preserve_keys = false])

- **Example**

```
<?php
```

```
$salary=array("Sonoo"=>"550000","Vimal"=>"250000","Ratan"=>"200000")  
print_r(array_chunk($salary,2));
```

```
?>
```

Output:

```
Array ( [0] => Array ( [0] => 550000 [1] => 250000 )  
[1] => Array ( [0] => 200000 ) )
```

Array Functions(3)

PHP count() function

- PHP count() function counts all elements in an array.
- **Syntax**

int count (mixed \$array_or_countable [, int \$mode = COUNT_NORMAL])

- **Example**

```
<?php
```

```
$season=array("summer","winter","spring","autumn");
```

```
echo count($season);
```

```
?>
```

Output:

4

Array Functions(4)

PHP sort() function

- PHP sort() function sorts all the elements in an array.
- **Syntax**

bool sort (**array** &\$array [, int \$sort_flags = SORT_REGULAR])

- **Example**

```
<?php
```

```
$season=array("summer","winter","spring","autumn");
```

```
sort($season);
```

```
foreach( $season as $s )
```

```
{
```

```
    echo "$s<br />";
```

```
}
```

```
?>
```

Output:

autumn

spring

summer

winter

Array Functions(5)

PHP array_reverse() function

- PHP array_reverse() function returns an array containing elements in reversed order.
- **Syntax**

array array_reverse (**array** \$array [, bool \$preserve_keys = false])

- **Example**

```
<?php
$season=array("summer","winter","spring","autumn");
$reverseseason=array_reverse($season);
foreach( $reverseseason as $s )
{
    echo "$s<br />";
}
?>
```

Output:

```
autumn
spring
winter
summer
```

Array Functions(6)

PHP array_search() function

- PHP array_search() function searches the specified value in an array. It returns key if search is successful.
- **Syntax**

mixed array_search (mixed \$needle , **array** \$haystack [, bool \$strict = false])

- **Example**

```
<?php
$season=array("summer","winter","spring","autumn");
$key=array_search("spring",$season);
echo $key;
?>
```

Output:

2

Array Functions(7)

PHP array_intersect() function

- PHP array_intersect() function returns the intersection of two array. In other words, it returns the matching elements of two array.

- **Syntax**

array array_intersect (**array** \$array1 , **array** \$array2 [, **array** \$...])

- **Example**

```
<?php
$name1=array("sonoo","john","vivek","smith");
$name2=array("umesh","sonoo","kartik","smith");
$name3=array_intersect($name1,$name2);
foreach( $name3 as $n )
{
    echo "$n<br />";
}
?>
```

Output:

sonoo
smith

PHP String

- PHP string is a sequence of characters i.e., used to store and manipulate text.
- PHP supports only 256-character set and so that it does not offer native Unicode support.
- There are various ways to specify a string literal in PHP.
 - **single quoted**
 - **double quoted**
 - **heredoc syntax**
 - **newdoc syntax (since PHP 5.3)**

PHP String

Single Quoted

- We can create a string in PHP by enclosing the text in a single-quote.
- It is the easiest way to specify string in PHP.
- For specifying a literal single quote, escape it with a backslash (\) and to specify a literal backslash (\) use double backslash (\\).
- **Example:**

```
<?php
```

```
    $str='Hello text within single quote';
```

```
    echo $str;
```

```
?>
```

Output:

Hello text within single quote

PHP String

Double Quoted

- In PHP, we can specify string through enclosing text within double quote also. But escape sequences and variables will be interpreted using double quote PHP strings.

- **Example 1**

```
<?php  
$str="Hello text within double quote";  
echo $str;  
?>
```

Output:

Hello text within double quote

String Function(1)

PHP strtoupper() function

- The strtoupper() function returns string in uppercase letter.

- **Syntax**

string strtoupper (string \$string)

- **Example**

```
<?php
$str="My name is abc";
$str=strtoupper($str);
echo $str;
?>
```

Output:

MY NAME IS ABC

String Function(2)

PHP ucfirst() function

- The ucfirst() function returns string converting first character into uppercase. It doesn't change the case of other characters.

- **Syntax**

string ucfirst (string \$str)

- **Example**

```
<?php
```

```
$str="my name is abc";
```

```
$str=ucfirst($str);
```

```
echo $str;
```

```
?>
```

Output:

My name is abc

String Function(3)

PHP ucwords() function

- The ucwords() function returns string converting first character of each word into uppercase.

- **Syntax**

string ucwords (string \$str)

- **Example**

```
<?php
$str="my name is Sonoo jaiswal";
$str=ucwords($str);
echo $str;
?>
```

Output:

My Name Is Sonoo Jaiswal

String Function(4)

PHP strrev() function

- The strrev() function returns reversed string.

- **Syntax**

string strrev (string \$string)

- **Example**

```
<?php
```

```
$str="my name is Sonoo jaiswal";
```

```
$str=strrev($str);
```

```
echo $str;
```

```
?>
```

Output:

lawsiaj oonoS si eman ym

String Function(5)

PHP strlen() function

- The strlen() function returns length of the string.

- **Syntax**

```
int strlen ( string $string )
```

- **Example**

```
<?php  
$str="my name is Sonoo jaiswal";  
$str= strlen($str);  
echo $str;  
?>
```

Output:

24

String: Example

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

<?php
echo strlen("Hello world!");
echo str_word_count("Hello world!");
echo strrev("Hello world!");
echo strpos("Hello world!", "world");
echo str_replace("world", "Dolly", "Hello world!");
?>

</body>
</html>
```


String Operator

Operator	Name	Example	Result
.	Concatenation	\$txt1 . \$txt2	Concatenation of \$txt1 and \$txt2
.=	Concatenation assignment	\$txt1 .= \$txt2	Appends \$txt2 to \$txt1

String Operator: Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<?php
```

```
$txt1 = "Hello";
```

```
$txt2 = " world!";
```

```
echo $txt1 . $txt2;
```

```
?>
```

```
</body>
```

```
</html>
```

HTML Forms

HTML Forms

- HTML Forms are required, when you **want to collect some data from the site visitor**.
- For example, during user registration you would like to collect information such as name, email address, credit card, etc.
- A form will **take input from the site visitor and then will post it to a back-end application** such as CGI, ASP Script or PHP script etc.
- There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

Forms Syntax

```
<form action = "Script URL" method = "GET|POST">  
form elements like input, textarea etc.  
</form>
```

action

- Backend script ready to process your passed data.

method

- Method to be used to upload data. The most frequently used are GET and POST methods.

Forms Syntax

Form elements

Text Input Controls , Checkboxes Controls, Radio Box Controls, Select Box Controls, File Select boxes , Hidden Controls, Clickable Buttons , Submit and Reset Button

target

- Specify the target window or frame where the result of the script will be displayed. It takes values like `_blank`, `_self`, `_parent` etc.

GET Method

- The default method when submitting form data is GET.
- However, when GET is used, the submitted form data will be **visible in the page address field**:

When to use GET??

- Appends form-data into the URL in name/value pairs
- The length of a URL is limited (about 3000 characters)
- Never use GET to send sensitive data! (will be visible in the URL)
- Useful for form submissions where a user wants to bookmark the result
- GET is better for non-secure data, like query strings in Google

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<p>This form will be submitted using the GET method:</p>
```

```
<form method="GET" target="_blank" >
```

```
  First name:<br>
```

```
  <input type="text" name="firstname" value="Mickey">
```

```
  <br>
```

```
  Last name:<br>
```

```
  <input type="text" name="lastname" value="Mouse">
```

```
  <br><br>
```

```
  <input type="submit" value="Submit">
```

```
</form>
```

```
</body>
```

```
</html>
```


POST Method

- Always use POST if the form data contains sensitive or personal information.
- The POST method does not display the submitted form data in the page address field.
- **When to use POST??**
 - POST has no size limitations, and can be used to send large amounts of data.
 - Form submissions with POST cannot be bookmarked

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<p>This form will be submitted using the POST method:</p>
```

```
<form method="POST" target="_blank" >
```

```
  First name:<br>
```

```
  <input type="text" name="firstname" value="Mickey">
```

```
  <br>
```

```
  Last name:<br>
```

```
  <input type="text" name="lastname" value="Mouse">
```

```
  <br><br>
```

```
  <input type="submit" value="Submit">
```

```
</form>
```

```
</body>
```

```
</html>
```

HTML Form Controls

There are different types of form controls that you can use to collect data using HTML form –


- Text Input Controls
- Checkboxes Controls
- Radio Box Controls
- Select Box Controls
- File Select boxes
- Hidden Controls
- Clickable Buttons
- Submit and Reset Button

Text Input Controls

- **Single-line text input controls** – This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML `<input>` tag.
- **Password input controls** – This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML `<input>` tag.
- **Multi-line text input controls** – This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML `<textarea>` tag.

1. Single-line text input controls

```
<!DOCTYPE html>
<html>
<head>
<title>Text Input Control</title>
</head>
<body>
<form > First name: <input type = "text" name = "first_name" />
  <br>
  Last name: <input type = "text" name = "last_name" />
</form>
</body> </html>
```



1. Single-line text input controls

Sr.No	Attribute & Description
1	type Indicates the type of input control and for text input control it will be set to text .
2	name Used to give a name to the control which is sent to the server to be recognized and get the value.
3	value This can be used to provide an initial value inside the control.
4	size Allows to specify the width of the text-input control in terms of characters.
5	maxlength Allows to specify the maximum number of characters a user can enter into the text box.

2. Password input controls

```
<!DOCTYPE html>
<html>
  <head>
    <title>Password Input Control</title>
  </head>
  <body>
    <form >
      User ID : <input type = "text" name = "user_id" /> <br>
      Password: <input type = "password" name = "password" />
    </form>
  </body> </html>
```

User ID :

Password:

2. Password input controls

Sr.No	Attribute & Description
1	type Indicates the type of input control and for password input control it will be set to password .
2	name Used to give a name to the control which is sent to the server to be recognized and get the value.
3	value This can be used to provide an initial value inside the control.
4	size Allows to specify the width of the text-input control in terms of characters.
5	maxlength Allows to specify the maximum number of characters a user can enter into the text box.

3. Multiple-Line Text Input Controls

```
<!DOCTYPE html>
<html>
<head>
<title>Multiple-Line Input Control</title>
</head>
<body>
<form> Description : <br />
  <textarea rows = "5" cols = "50" name = "description"> Enter
    description here... </textarea>
</form>
</body> </html>
```

Description:

Enter description here...

3. Multiple-Line Text Input Controls

Sr.No	Attribute & Description
1	name Used to give a name to the control which is sent to the server to be recognized and get the value.
2	rows Indicates the number of rows of text area box.
3	cols Indicates the number of columns of text area box

Checkbox Control

- Checkboxes are used when more than one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to **checkbox**..

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Checkbox Control</title>
```

```
</head>
```

```
<body>
```

```
<form>
```

```
<input type = "checkbox" name = "maths" value = "on"> Maths
```

```
<input type = "checkbox" name = "physics" value = "on"> Physics
```

```
</form>
```

```
</body> </html>
```

☐ Maths ☐ Physics

Checkbox Control

Sr.No	Attribute & Description
1	type Indicates the type of input control and for checkbox input control it will be set to checkbox .
2	name Used to give a name to the control which is sent to the server to be recognized and get the value.
3	value The value that will be used if the checkbox is selected.
4	checked Set to <i>checked</i> if you want to select it by default.

Radio Button Control

- Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML `<input>` tag but type attribute is set to **radio**.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Radio Box Control</title>
```

```
</head>
```

```
<body>
```

```
<form> <input type = "radio" name = "subject" value = "maths">  
  Maths
```

```
<input type = "radio" name = "subject" value = "physics">  
  Physics
```

```
</form> </body> </html>
```

☐ Maths ☒ Physics

Radio Button Control

Sr.No	Attribute & Description
1	type Indicates the type of input control and for checkbox input control it will be set to radio.
2	name Used to give a name to the control which is sent to the server to be recognized and get the value.
3	value The value that will be used if the radio box is selected.
4	checked Set to <i>checked</i> if you want to select it by default.

Select Box Control

- A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Select Box Control</title>
```

```
</head>
```

```
<body>
```

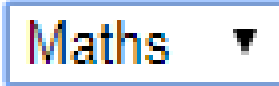
```
<form>
```

```
<select name = "dropdown
```

```
<option value = "Maths" selected
```

```
<option value = "Physics">Physics</option> </select>
```

```
</form> </body> </html>
```



Select Box Control

Sr.No	Attribute & Description
1	name Used to give a name to the control which is sent to the server to be recognized and get the value.
2	size This can be used to present a scrolling list box.
3	multiple If set to "multiple" then allows a user to select multiple items from the menu.

Select Box Control

Sr.No	Attribute & Description
1	value The value that will be used if an option in the select box box is selected.
2	selected Specifies that this option should be the initially selected value when the page loads.
3	label An alternative way of labeling options

File Upload Box

- If you want to allow a user to upload a file to your web site, you will need to use a file upload box, also known as a file select box. This is also created using the `<input>` element but type attribute is set to **file**.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> <title>File Upload Box</title> </head>
```

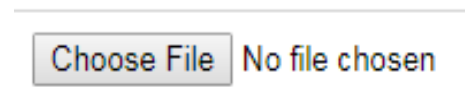
```
<body>
```

```
<form>
```

```
<input type = "file" name = "fileupload" accept = "image/*" />
```

```
</form>
```

```
</body> </html>
```



File Upload Box

Sr.No	Attribute & Description
1	name Used to give a name to the control which is sent to the server to be recognized and get the value.
2	accept Specifies the types of files that the server accepts.

Button Controls

- There are various ways in HTML to create clickable buttons. You can also create a clickable button using `<input>` tag by setting its type attribute to **button**.

Sr.No	Type & Description
1	submit This creates a button that automatically submits a form.
2	reset This creates a button that automatically resets form controls to their initial values.
3	button This creates a button that is used to trigger a client-side script when the user clicks that button.
4	image This creates a clickable button but we can use an image as background of the button.

Button Controls

```
<!DOCTYPE html>
<html>
<head> </head>
<body>
  <form>
    <input type = "submit" name = "submit" value = "Submit" />
    <input type = "reset" name = "reset" value = "Reset" />
    <input type = "button" name = "ok" value = "OK" />
    <input type = "image" name = "imagebutton" src =
      "/html/images/logo.png" />
  </form>
</body> </html>
```

The Target Attribute

- The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window.
- The default value is "_self" which means the form will be submitted in the current window.
- To make the form result open in a new browser tab, use the value "_blank":

The Target Attribute

```
<!DOCTYPE html>
```

```
<html> <body>
```

```
<p>When submitting this form, the result will be opened in a new browser  
tab:</p>
```

```
<form target="_blank">
```

```
First name:<br>
```

```
<input type="text" name="firstname" value="Mickey">
```

```
<br>
```

```
Last name:<br>
```

```
<input type="text" name="lastname" value="Mouse">
```

```
<br><br>
```

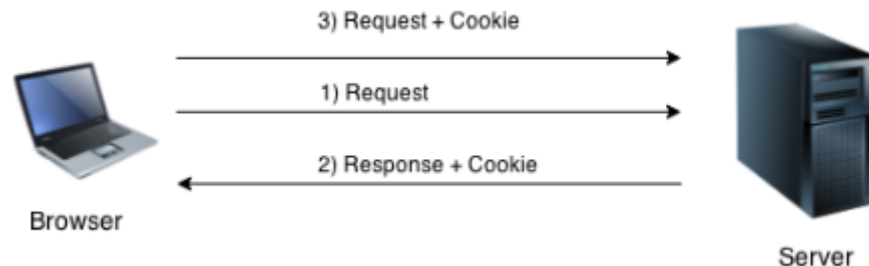
```
<input type="submit" value="Submit">
```

```
</form>
```

```
</body> </html>
```

PHP Cookie

- PHP cookie is a small piece of information which is stored at client browser. It is used to recognize the user.
- Cookie is created at server side and saved to client browser.
- Each time when client sends request to the server, cookie is embedded with request. Such way, cookie can be received at the server side.
- In short, cookie can be created, sent and received at server end.



PHP Cookie (contd..)

PHP setcookie() function

- PHP setcookie() function is used to set cookie with HTTP response. Once cookie is set, you can access it by \$_COOKIE superglobal variable.

- **Syntax**

```
bool setcookie ( string $name [, string $value [, int $expire = 0 [, string $path [, string $domain [, bool $secure = false [, bool $httponly = false ]]]]] )
```

- **Example**

```
setcookie("CookieName", "CookieValue");  
/* defining name and value only */  
setcookie("CookieName", "CookieValue", time()+1*60*60);  
//using expiry in 1 hour(1*60*60 seconds or 3600 seconds)  
setcookie("CookieName", "CookieValue", time()+1*60*60, "/mypath/", "  
mydomain.com", 1);
```

PHP Cookie (contd..)

PHP \$_COOKIE

- PHP \$_COOKIE superglobal variable is used to get cookie.
- **Example**

`$value=$_COOKIE["CookieName"];`//returns cookie value

Example: Cookie

```
<?php
setcookie("user", "Sonoo");
?>
<html>
<body>
<?php
if(!isset($_COOKIE["user"])) {
    echo "Sorry, cookie is not found!";
}
else
{
    echo "<br/>Cookie Value: " . $_COOKIE["user"];
}
?>
</body>
</html>
```

Output:

Sorry, cookie is not found!

Firstly cookie is not set. But, if you *refresh* the page, you will see cookie is set now.

Cookie Value: Sonoo

PHP Cookie (contd..)

PHP Delete Cookie

- If you set the expiration date in past, cookie will be deleted.
- Example:

```
<?php
```

```
setcookie ("CookieName", "", time() -3600);
```

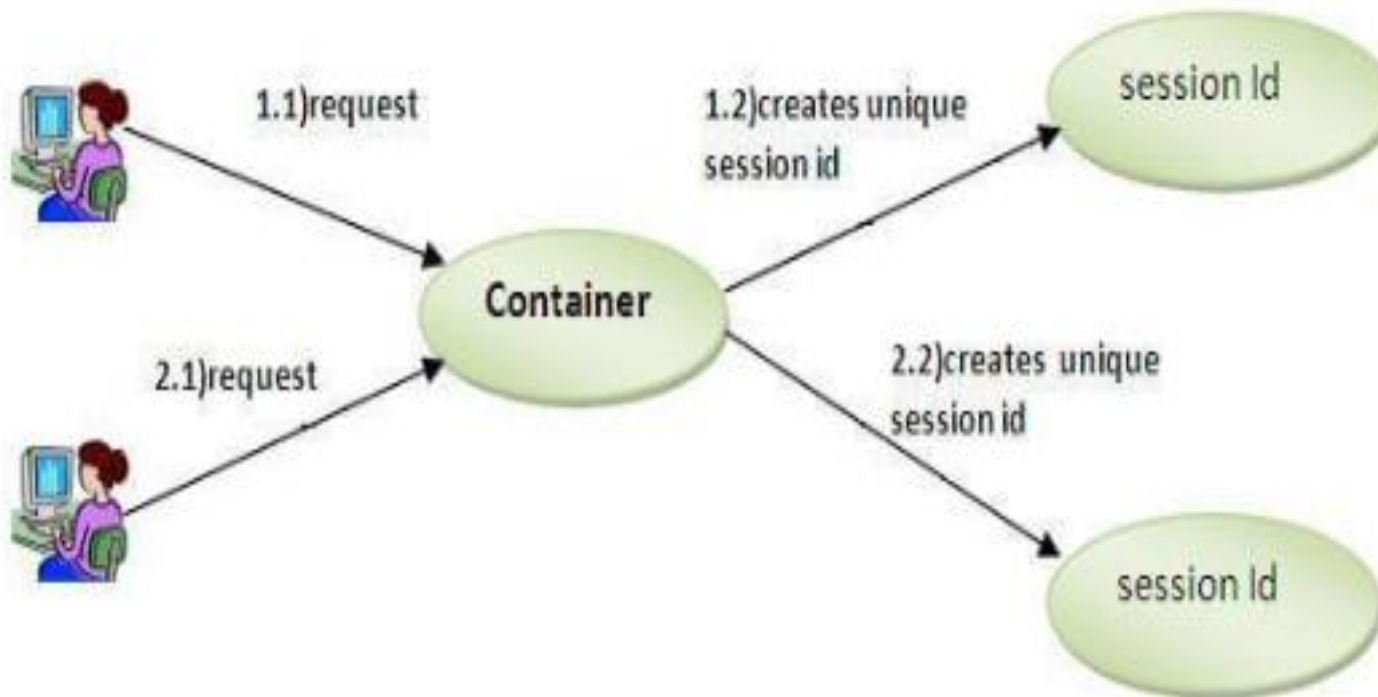
```
// set the expiration date to one hour ago
```

```
?>
```

PHP Session

- PHP session is used to store and pass information from one page to another temporarily (until user close the website).
- PHP session technique is widely used in shopping websites where we need to store and pass cart information e.g. username, product code, product name, product price etc from one page to another.
- PHP session creates unique user id for each browser to recognize the user and avoid conflict between multiple browsers.

PHP Session



PHP Session

PHP session_start() function

- PHP session_start() function is used to start the session. It starts a new or resumes existing session. It returns existing session if session is created already. If session is not available, it creates and returns new session.

- **Syntax**

`bool session_start (void)`

- **Example**

`session_start();`

PHP Session

PHP \$_SESSION

- PHP \$_SESSION is an associative array that contains all session variables. It is used to set and get session variable values.

- **Example: Store information**

```
$_SESSION["user"] = "Sachin";
```

- **Example: Get information**

```
echo $_SESSION["user"];
```


Example: PHP Session

Session1.php

```
<?php
session_start();
?>
<html>
<body>
<?php
$_SESSION["user"] = "Sachin";
echo "Session information are set successfuly.<br/>";
?>
<a href="session2.php">Visit next page</a>

</body>
</html>
```

Session2.php

```
<?php
session_start();
?>
<html>
<body>
<?php
echo "User is: " . $_SESSION["user"];
?>
</body>
</html>
```

PHP Session Counter Example

```
<?php
    session_start();

    if (!isset($_SESSION['counter'])) {
        $_SESSION['counter'] = 1;
    } else {
        $_SESSION['counter']++;
    }
    echo ("Page Views: " . $_SESSION['counter']);
?>
```

PHP Session

PHP Destroying Session

- PHP `session_destroy()` function is used to destroy all session variables completely.

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

PHP File Handling

- PHP File System allows us to create file, read file line by line, read file character by character, write file, append file, delete file and close file.
- **PHP Open File - fopen()**
- The PHP fopen() function is used to open a file
- **Syntax**

resource fopen (string \$filename , string \$mode [, bool \$use_include_path = false [, resource \$context]])

- **Example**

```
<?php
$handle = fopen("c:\\folder\\file.txt", "r");
?>
```

PHP File Handling

PHP Close File - fclose()

- The PHP fclose() function is used to close an open file pointer.

- **Syntax**

ool fclose (resource \$handle)

- **Example**

```
<?php
```

```
fclose($handle);
```

```
?>
```

PHP File Handling

PHP Read File - fread()

- The PHP fread() function is used to read the content of the file. It accepts two arguments: resource and file size.

- **Syntax**

string fread (resource \$handle , int \$length)

- **Example**

```
<?php
```

```
$filename = "c:\\myfile.txt";
```

```
$handle = fopen($filename, "r");//open file in read mode
```

```
$contents = fread($handle, filesize($filename));//read file
```

```
echo $contents;//printing data of file
```

```
fclose($handle);//close file
```

```
?>
```

Output:

hello php file

PHP File Handling

PHP Write File - fwrite()

- The PHP fwrite() function is used to write content of the string into file.

- **Syntax**

int fwrite (resource \$handle , string \$string [, int \$length])

- **Example**

```
<?php
```

```
$fp = fopen('data.txt', 'w');//open file in write mode
```

```
fwrite($fp, 'hello ');
```

```
fwrite($fp, 'php file');
```

```
fclose($fp);
```

```
echo "File written successfully";
```

```
?>
```

Output:

File written successfully

PHP File Handling

PHP Delete File - unlink()

- The PHP unlink() function is used to delete file.

- **Syntax**

bool unlink (string \$filename [, resource \$context])

- **Example**

```
<?php
```

```
unlink('data.txt');
```

```
echo "File deleted successfully";
```

```
?>
```


phpMyAdmin

- phpMyAdmin is an open-source software tool which is written in PHP to manage the tables and data inside the database.
- The main purpose of phpMyAdmin is to **handle the administration of MySQL over the web.**
- We can create, update, drop, alter, delete, import, and export MySQL database tables by using this software.
- phpMyAdmin also supports a wide range of operation like **managing databases, relations, tables, columns, indexes, permissions, and users**, etc., on MySQL and MariaDB.

These operations can be performed via user interface, while we still have the ability to execute any SQL statement.

- phpMyAdmin can also be used to perform administrative tasks such as **database creation, query execution.**

phpMyAdmin

Prerequisite

- Web server - Apache, Nginx, IIS
- PHP
- Database - MySQL, MariaDB
- Web Browser



Bringing MySQL to the web

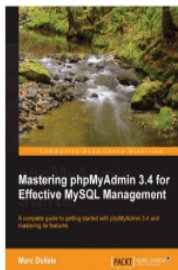
Download 4.9.0.1

Try demo

Donate

About

phpMyAdmin is a free software tool written in [PHP](#), intended to handle the administration of [MySQL](#) over the Web. phpMyAdmin supports a wide range of operations on MySQL and MariaDB. Frequently used operations (managing databases, tables, columns, relations, indexes, users, permissions, etc) can be performed via the user interface, while you still have the ability to directly execute any SQL statement.



phpMyAdmin comes with a wide range of [documentation](#) and users are welcome to update [our wiki pages](#) to share ideas and howtos for various operations. The [phpMyAdmin team](#) will try to help you if you face any problem; you can use a [variety of support channels](#) to get help.

phpMyAdmin is also very deeply documented in a book written by one of the developers – [Mastering phpMyAdmin for Effective MySQL Management](#), which is available in English and Spanish.

To ease usage to a wide range of people, phpMyAdmin is being translated into [72 languages](#) and supports both LTR and RTL languages.

phpMyAdmin is a mature project with a stable and flexible code base; you can find out more about the [project and its history](#) and the [awards](#) it earned. When the project turned 15, we published a [celebration page](#).

The phpMyAdmin project is a member of [Software Freedom Conservancy](#). SFC is a not-for-profit organization that helps promote, improve, develop, and defend Free, Libre, and Open Source Software (FLOSS) projects.

Sponsors

Diamond sponsor

This space is available — contact us to get listed here.

Platinum sponsors

Thank you for downloading phpMyAdmin ✕

Your download should start soon, if not please [click here](#).

Please verify the downloaded file

Please take additional steps to verify that the file you have downloaded is not corrupted, you can verify it using the following methods:

- Verify its [PGP signature](#), see the [Verifying phpMyAdmin releases](#) chapter for more information.
- Check that the file's SHA256 hash matches

`04ef793d42c70e891429de134d91de38464801abe195022786486c3e88c9e403`

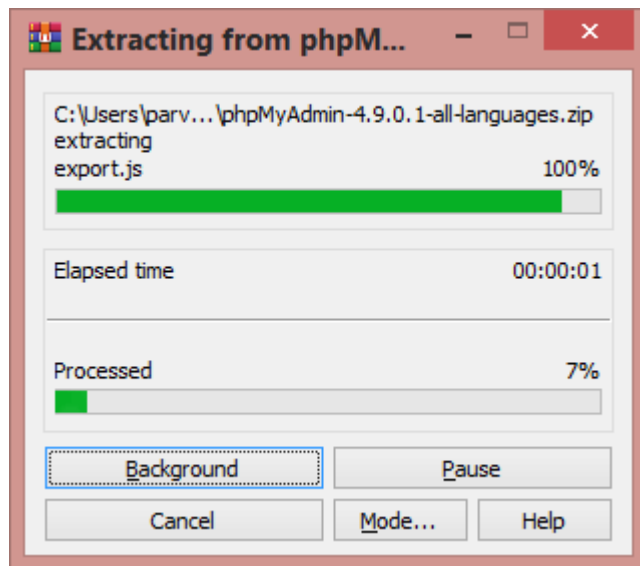
phpMyAdmin needs your continued support to grow and thrive

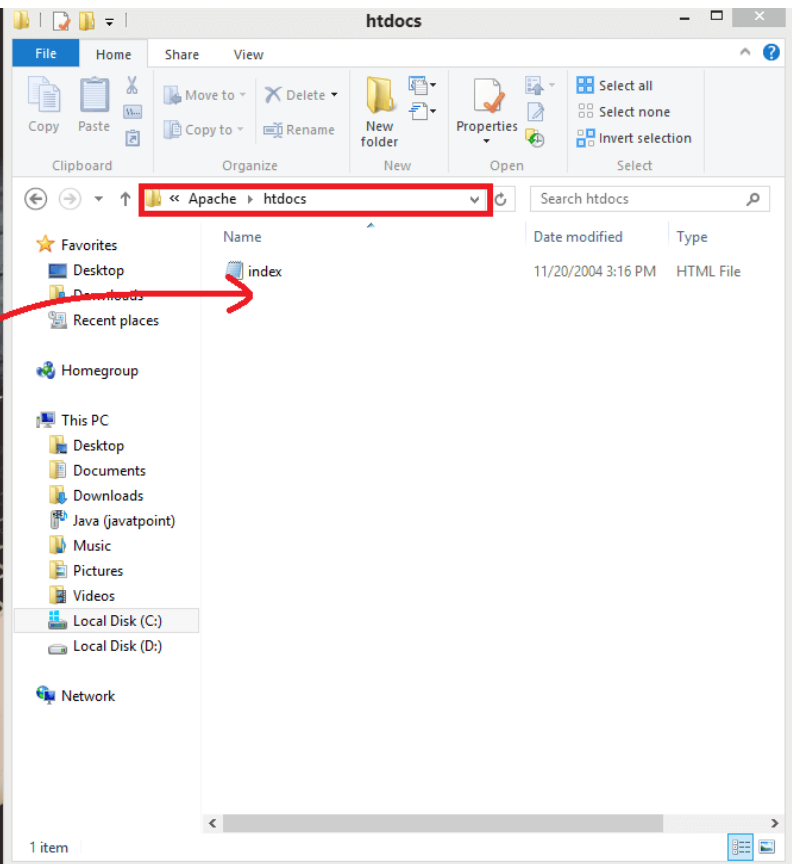
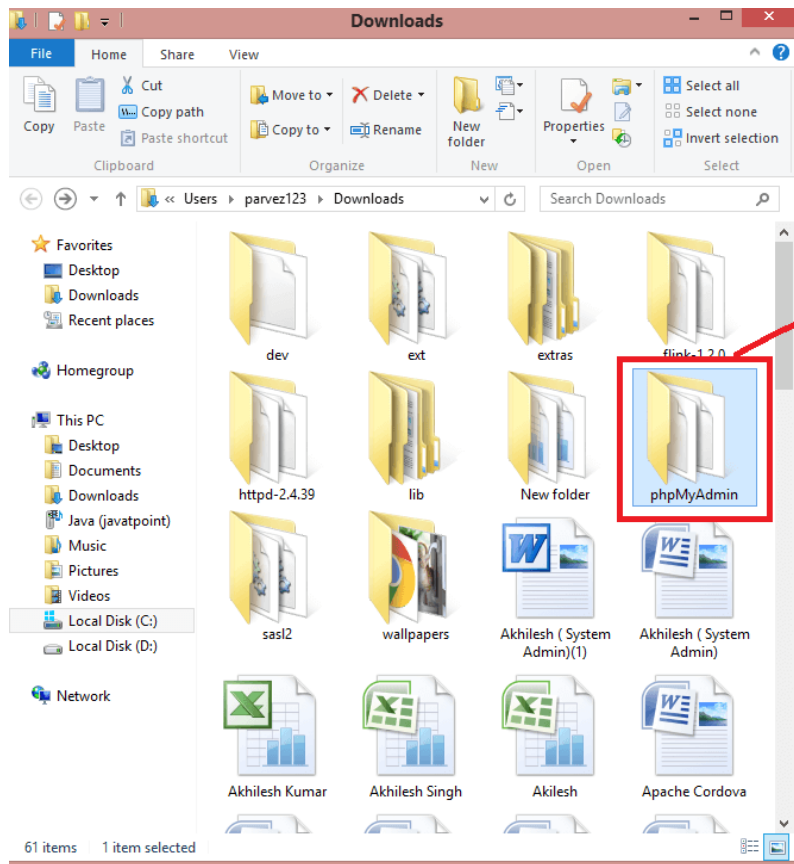
phpMyAdmin would not exist without the work of many volunteers and contractors. You can support us to make phpMyAdmin even better by [donating to our project](#). Every donation counts!

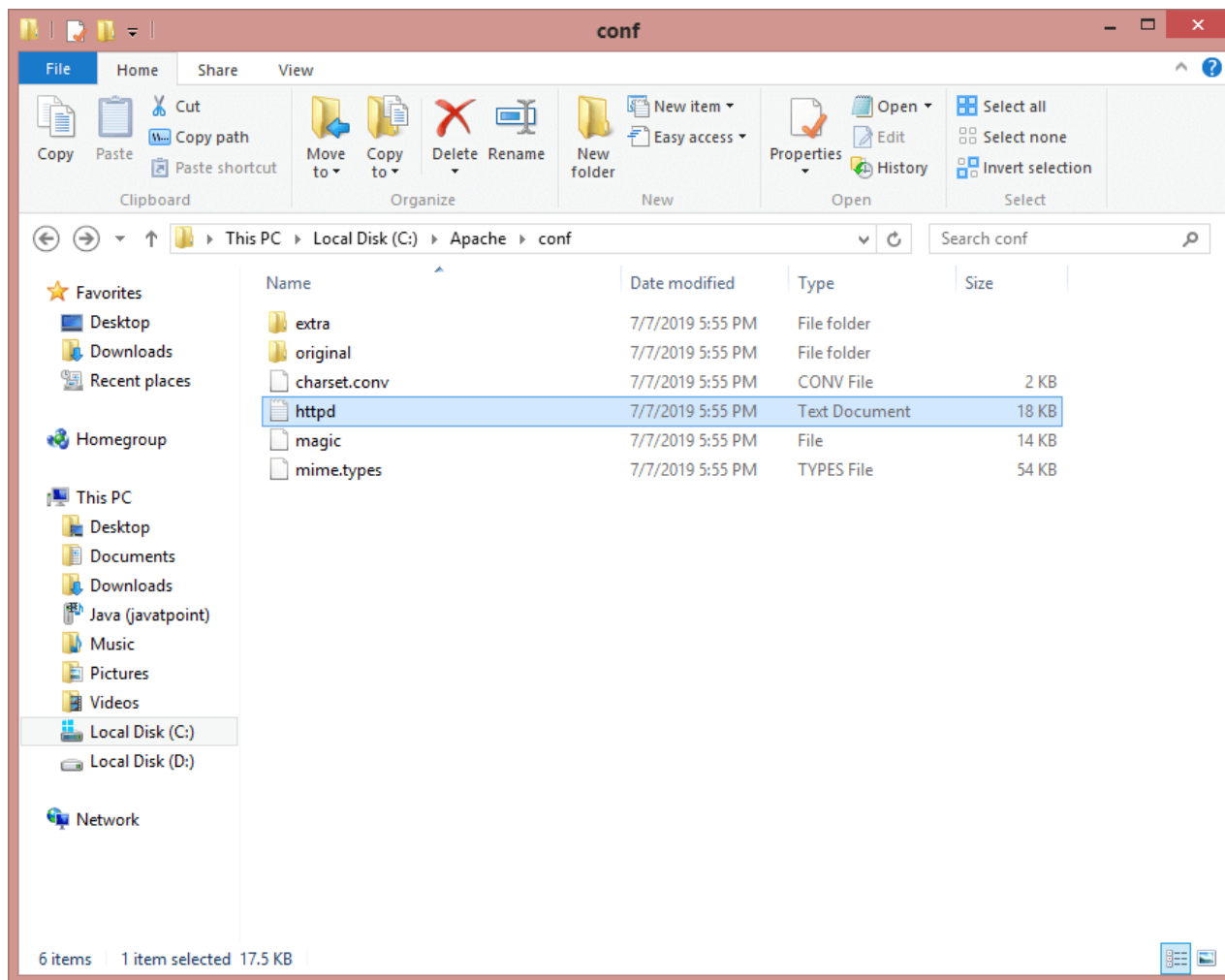
We have also a [sponsorship program](#) for corporates who are willing to spent more money and get some benefits such as a logo placement in return.

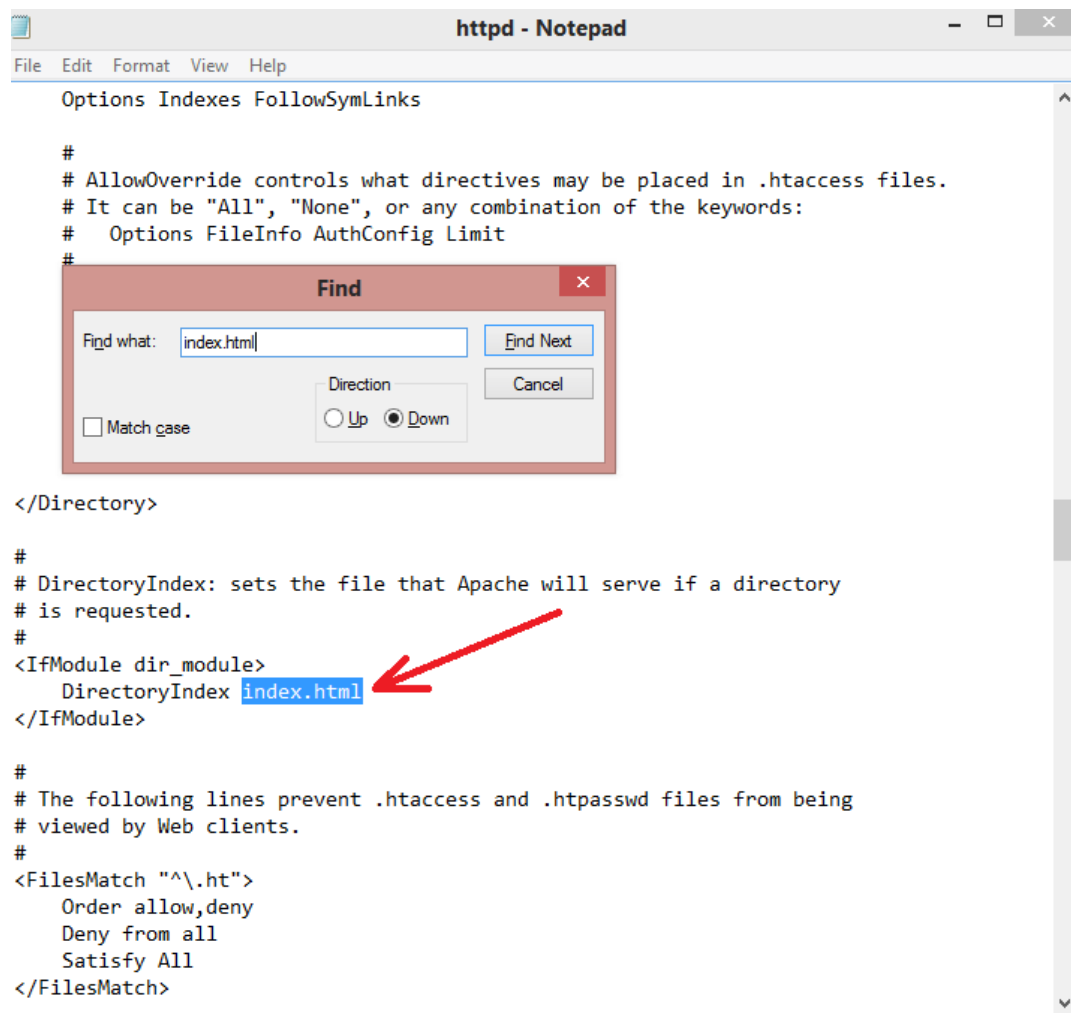
Close

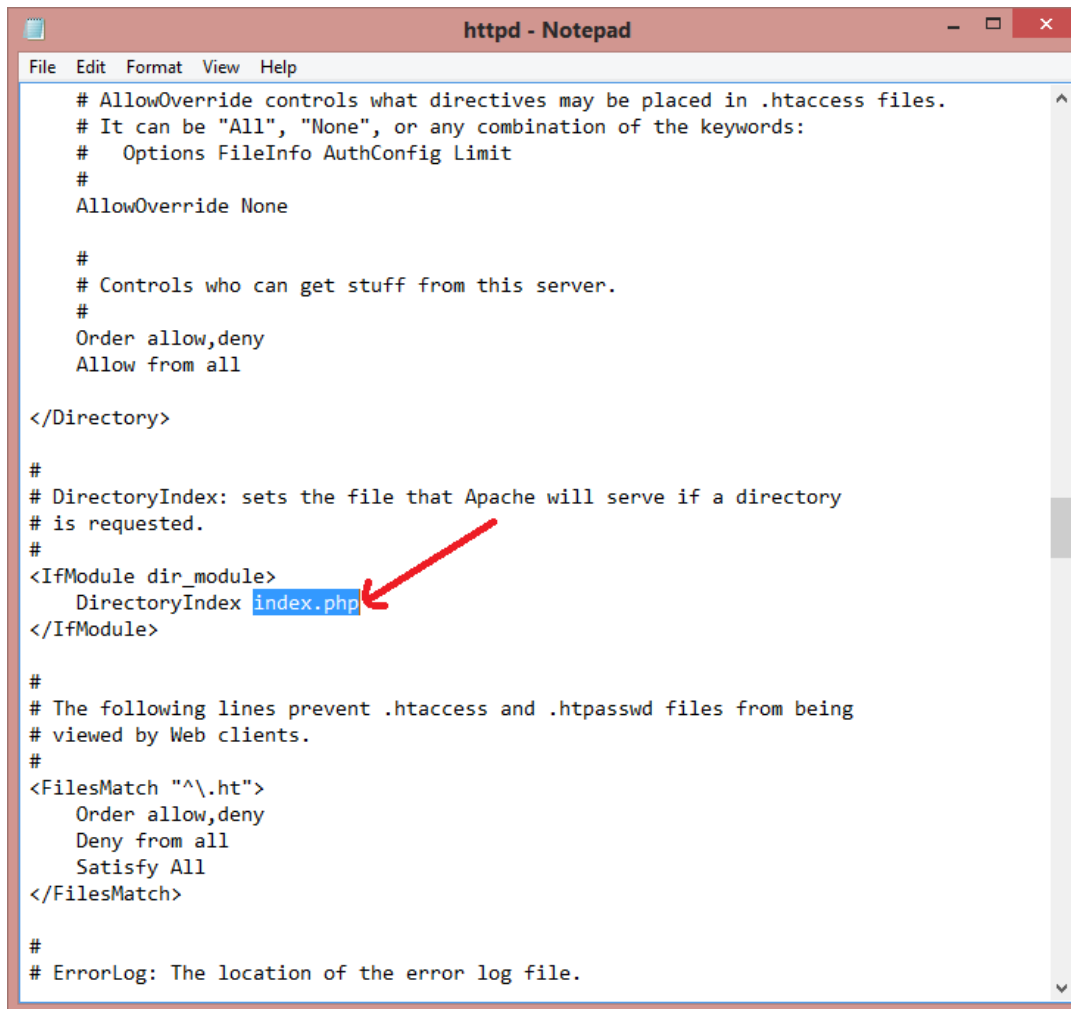
Donate to phpMyAdmin











```
File Edit Format View Help

# AllowOverride controls what directives may be placed in .htaccess files.
# It can be "All", "None", or any combination of the keywords:
#   Options FileInfo AuthConfig Limit
#
AllowOverride None

#
# Controls who can get stuff from this server.
#
Order allow,deny
Allow from all

</Directory>

#
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
#
<IfModule dir_module>
    DirectoryIndex index.php
</IfModule>

#
# The following lines prevent .htaccess and .htpasswd files from being
# viewed by Web clients.
#
<FilesMatch "^\.ht">
    Order allow,deny
    Deny from all
    Satisfy All
</FilesMatch>

#
# ErrorLog: The location of the error log file.
```


localhost / 127.0.0.1 | phpMyAdmin x

localhost/phpmyadmin/

phpMyAdmin

Recent Favorites

- New
- information_schema
- mysql
- performance_schema
- phpmyadmin
- test

Server: 127.0.0.1

Databases SQL Status User accounts Export Import Settings Replication Variables Charsets More

General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

Font size: 82%

More settings

Database server

- Server: 127.0.0.1 via TCP/IP
- Server type: MariaDB
- Server connection: SSL is not being used
- Server version: 10.3.16-MariaDB - mariadb.org binary distribution
- Protocol version: 10
- User: root@localhost
- Server charset: cp1252 West European (latin1)

Web server

- Apache/2.4.39 (Win64) OpenSSL/1.0.2s PHP/7.1.30
- Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - \$Id: 38fea24f2847fa7519001be390c98ae0acafe387\$
- PHP extension: mysqli curl mbstring
- PHP version: 7.1.30

phpMyAdmin

- Version information: 4.9.0.1 (up to date)
- Documentation
- Official Homepage
- Contribute
- Get support
- List of changes

Console

localhost/phpmyadmin/server_databases.php?server=1

phpMyAdmin

Recent Favorites

- New
- information_schema
- mysql
- performance_schema
- phpmyadmin
- test

Server: 127.0.0.1

Databases SQL Status User accounts Export Import Settings Replication Variables Charsets More

Databases

Create database

Student

latin1_swedish_ci

Create

Database	Collation	Action
<input type="checkbox"/> information_schema	utf8_general_ci	Check privileges
<input type="checkbox"/> mysql	latin1_swedish_ci	Check privileges
<input type="checkbox"/> performance_schema	utf8_general_ci	Check privileges
<input type="checkbox"/> phpmyadmin	utf8_bin	Check privileges
<input type="checkbox"/> test	latin1_swedish_ci	Check privileges
Total: 5		latin1_swedish_ci

☐ Check all With selected: [Drop](#)

Note: Enabling the database statistics here might cause heavy traffic between the web server and the MySQL server.

- Enable statistics

Console

← → ↻ localhost/phpmyadmin/server_databases.php?server=1

phpMyAdmin

Recent Favorites

- New
- information_schema
- mysql
- newdb
- performance_schema
- phpmyadmin
- student
- test

Server: 127.0.0.1 » Database: newdb

Structure SQL Search Query Export Import Operations Privileges Routines Events More

⚠ No tables found in database.

Create table

Name: Number of columns:

Go

Console

localhost/phpmyadmin/db_structure.php?server=1&db=newdb

Server: 127.0.0.1 » Database: newdb » Table: Employee

Browse Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers

Table name: Employee Add 1 column(s) Go

Name	Type	Length/Values	Default	Collation	Attributes	Null	Index
EmpID	INT		None			<input type="checkbox"/>	---
<small>Pick from Central Columns</small>							
Name	VARCHAR	40	None			<input type="checkbox"/>	---
<small>Pick from Central Columns</small>							
Address	VARCHAR	100	None			<input type="checkbox"/>	---
<small>Pick from Central Columns</small>							
Mobile Number	VARCHAR	10	None			<input type="checkbox"/>	---
<small>Pick from Central Columns</small>							
Gender	VARCHAR	6	None			<input type="checkbox"/>	---
<small>Pick from Central Columns</small>							

Table comments: Collation: Storage Engine: InnoDB


PARTITION definition:

Partition by: (Expression or column list)

Partitions:

Console

Preview SQL Save



localhost/phpmyadmin/tbl_structure.php?server=1&db=newdb&table=employee

Server: 127.0.0.1 » Database: newdb » Table: employee

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	EmpID	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	Name	varchar(40)	latin1_swedish_ci		No	None			Change Drop More
3	Address	varchar(100)	latin1_swedish_ci		No	None			Change Drop More
4	Mobile Number	varchar(10)	latin1_swedish_ci		No	None			Change Drop More
5	Gender	varchar(6)	latin1_swedish_ci		No	None			Change Drop More

☐ Check all With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#)
[Remove from central columns](#)

[Print](#) [Propose table structure](#) [Track table](#) [Move columns](#) [Normalize](#)

[Add](#) 1 column(s) after Gender [Go](#)

Indexes

Action	Keyname	Type	Unique	Packed	Column	Cardinality	Collation	Null	Comment
Edit Drop	PRIMARY	BTREE	Yes	No	EmpID	0	A	No	

Create an index on 1 columns [Go](#)

Partitions

No partitioning defined!

Console

Database creation using coding with phpMyAdmin

```
<?php
$servername = "localhost";
$username = "root"; //default user name is root
$password = ""; //default password is blank
$conn = mysqli_connect($servername, $username, $password);
if(!$conn)
    die("Connection failed".mysqli_connect_error());
else
    //echo "Successfully connected with database";
$query = "CREATE DATABASE newDB";
if (mysqli_query($conn, $query)) {
    echo "Database created successfully with the name newDB";
} else {
    echo "Error creating database: " . mysqli_error($conn);
}
mysqli_close($conn);
?>
```

Output:

Database created successfully with the name newDB.

← → ↻ ⓘ localhost/xampp/PMA/connection.php

Database created successfully with the name newDB

Database connectivity with phpMyAdmin

```
<?php
$dbhost="localhost";
$dbName="newDB";
$user="root";
$pass="";
$conn = new mysqli("mysql:host=$dbhost;dbname=$dbName",$user,$pass);

try{
    echo "Successfully connected with newdb database";
}
catch(Exception $e){
    die("Connection failed".$e->getMessage());
}
?>
```

Output:

Successfully connected with the database.

localhost/xampp/PMA/connection.php

Successfully connected with database