PHP
Examples

W3Schools Tutorial

Collected by:

Mohammad Samadi Gharajeh

www.msamadi.info

PHP Syntax

Write text to the output using PHP

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

<h1>My first PHP page</h1>

<?php
echo "Hello World!";
?>

</body>
</html>
```

Result:

My first PHP page

Hello World!

Add comments in PHP

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

Result:

10

Keywords, classes, functions, and user-defined functions ARE NOT case-sensitive

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

<?php
ECHO "Hello World!<br>";
echo "Hello World!<br>";
EcHo "Hello World!<br>";
?>

</body>
</html>
```

Result:

Hello World!
Hello World!
Hello World!

Variable names ARE case-sensitive

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

```
<?php
$color = "red";
echo "My car is " . $color . "<br>";
echo "My house is " . $COLOR . "<br>";
echo "My boat is " . $coLOR . "<br>";
?>

</body>
</html>
```

Result:

My car is red
My house is
My boat is

PHP Variables

Create different 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>
```

Result:

Hello world!
5
10.5

Test global scope (variable outside function)

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

<?php
$x = 5; // global scope

function myTest() {
    // using x inside this function will generate an error
```

```
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

echo "<p>Variable x outside function is: $x</p>";
?>

</body>
</html>
```

Result:

Variable x inside function is:

Variable x outside function is: 5

Test local scope (variable inside function)

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

<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();

// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>

</body>
</html>
```

Result:

Variable x inside function is: 5

Variable x outside function is:

Use the global keyword to access a global variable from within a function

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

<?php
$x = 5;
$y = 10;

function myTest() {
    global $x, $y;
    $y = $x + $y;
}

myTest(); // run function
echo $y; // output the new value for variable $y
?>

</body>
</html>
```

Result:

15

Use the $GLOBALS[] array to access a global variable from within a function

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

<?php
$x = 5;
$y = 10;

function myTest() {
    $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];
}
```

```
myTest();
echo $y;
?>

</body>
</html>
```

Result:

15

Use the static keyword to let a local variable not be deleted after execution of function

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

<?php
function myTest() {
    static $x = 0;
    echo $x;
    $x++;
}

myTest();
echo "<br>";
myTest();
echo "<br>";
myTest();
?>

</body>
</html>
```

Result:

0
1
2

PHP Echo and Print

Display strings with the echo command

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

<?php
echo "<h2>PHP is Fun!</h2>";
echo "Hello world!<br>";
echo "I'm about to learn PHP!<br>";
echo "This ", "string ", "was ", "made ", "with multiple parameters.";
?>

</body>
</html>
```

Result:

PHP is Fun!
Hello world!
I'm about to learn PHP!
This string was made with multiple parameters.

Display strings and variables with the echo command

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

<?php
$txt1 = "Learn PHP";
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

echo "<h2>" . $txt1 . "</h2>";
echo "Study PHP at " . $txt2 . "<br>";
echo $x + $y;
```

```
?>

</body>
</html>
```

Result:

Learn PHP
Study PHP at W3Schools.com
9

Display strings with the print command

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

<?php
print "<h2>PHP is Fun!</h2>";
print "Hello world!<br>";
print "I'm about to learn PHP!";
?>

</body>
</html>
```

Result:

PHP is Fun!
Hello world!
I'm about to learn PHP!

Display strings and variables with the print command

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

<?php
$txt1 = "Learn PHP";
```

```
$txt2 = "W3Schools.com";
$x = 5;
$y = 4;

print "<h2>" . $txt1 . "</h2>";
print "Study PHP at " . $txt2 . "<br>";
print $x + $y;
?>

</body>
</html>
```

Result:

Learn PHP
Study PHP at W3Schools.com
9

PHP Data Types

PHP string

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

<?php
$x = "Hello world!";
$y = 'Hello world!';

echo $x;
echo "<br>";
echo $y;
?>

</body>
</html>
```

Result:

Hello world!
Hello world!

PHP integer

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

<?php
$x = 5985;
var_dump($x);
?>

</body>
</html>
```

Result:

int(5985)

PHP float

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

<?php
$x = 10.365;
var_dump($x);
?>

</body>
</html>
```

Result:

float(10.365)

PHP array

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

<?php
$cars = array("Volvo","BMW","Toyota");
var_dump($cars);
?>

</body>
</html>
```

Result:

array(3) { [0]=> string(5) "Volvo" [1]=> string(3) "BMW" [2]=> string(6) "Toyota" }

PHP object

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

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

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

</body>
</html>
```

Result:

VW

PHP NULL value

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

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

</body>
</html>
```

Result:

NULL

PHP Strings

Get the length of a string - strlen()

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

<?php
echo strlen("Hello world!");
?>

</body>
</html>
```

Result:

12

Count the number of words in a string - str_word_count()

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

<?php
echo str_word_count("Hello world!");
?>

</body>
</html>
```

Result:

2

Reverse a string - strrev()

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

<?php
echo strrev("Hello world!");
?>

</body>
</html>
```

Result:

!dlrow olleH

Search for a specific text within a string - strpos()

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

<?php
echo strpos("Hello world!", "world");
?>

</body>
</html>
```

Result:

6

Replace text within a string - str_replace()

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

```
<?php
echo str_replace("world", "Dolly", "Hello world!");
?>

</body>
</html>
```

Result:

Hello Dolly!

PHP Constants

Case-sensitive constant name

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

<?php
// case-sensitive constant name
define("GREETING", "Welcome to W3Schools.com!");
echo GREETING;
?>

</body>
</html>
```

Result:

Welcome to W3Schools.com!

Case-insensitive constant name

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

<?php
// case-insensitive constant name
define("GREETING", "Welcome to W3Schools.com!", true);
echo greeting;
?>

</body>
</html>
```

Result:

Welcome to W3Schools.com!

PHP Operators

Arithmetic operator: Addition (+)

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

<?php
$x = 10;
$y = 6;

echo $x + $y;
?>

</body>
</html>
```

Result:

16

Arithmetic operator: Subtraction (-)

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

<?php
$x = 10;
$y = 6;

echo $x - $y;
?>

</body>
</html>
```

Result:

4

Arithmetic operator: Multiplication (*)

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

<?php
$x = 10;
$y = 6;

echo $x * $y;
?>

</body>
</html>
```

Result:

60

Arithmetic operator: Division (/)

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

<?php
$x = 10;
$y = 6;

echo $x / $y;
?>

</body>
</html>
```

Result:

1.6666666666667

Arithmetic operator: Modulus (%)

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

<?php
$x = 10;
$y = 6;

echo $x % $y;
?>

</body>
</html>
```

Result:

4

Assignment operator: x = y

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

<?php
$x = 10;
echo $x;
?>

</body>
</html>
```

Result:

10

Assignment operator: x += y

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

<?php
$x = 20;
$x += 100;

echo $x;
?>

</body>
</html>
```

Result:

120

Assignment operator: x -= y

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

<?php
$x = 50;
$x -= 30;

echo $x;
?>

</body>
</html>
```

Result:

20

*Assignment operator: x *= y*

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

<?php
$x = 5;
$x *= 6;

echo $x;
?>

</body>
</html>
```

Result:

30

Assignment operator: x /= y

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

<?php
$x = 10;
$x /= 5;

echo $x;
?>

</body>
</html>
```

Result:

2

Assignment operator: x %= y

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

<?php
$x = 15;
$x %= 4;

echo $x;
?>

</body>
</html>
```

Result:

3

Comparison operator: Equal (==)

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

<?php
$x = 100;
$y = "100";

var_dump($x == $y); // returns true because values are equal
?>

</body>
</html>
```

Result:

bool(true)

Comparison operator: Identical (===)

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

<?php
$x = 100;
$y = "100";

var_dump($x === $y); // returns false because types are not equal
?>

</body>
</html>
```

Result:

bool(false)

Comparison operator: Not equal (!=)

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

<?php
$x = 100;
$y = "100";

var_dump($x != $y); // returns false because values are equal
?>

</body>
</html>
```

Result:

bool(false)

Comparison operator: Not equal (<>)

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

<?php
$x = 100;
$y = "100";

var_dump($x <> $y); // returns false because values are equal
?>

</body>
</html>
```

Result:

bool(false)

Comparison operator: Not identical (!==)

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

<?php
$x = 100;
$y = "100";

var_dump($x !== $y); // returns true because types are not equal
?>

</body>
</html>
```

Result:

bool(true)

Comparison operator: Greater than (>)

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

<?php
$x = 100;
$y = 50;

var_dump($x > $y); // returns true because $x is greater than $y
?>

</body>
</html>
```

Result:

bool(true)

Comparison operator: Less than (<)

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

<?php
$x = 10;
$y = 50;

var_dump($x <  $y); // returns true because $x is less than $y
?>

</body>
</html>
```

Result:

bool(true)

Comparison operator: Greater than or equal (>=)

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

<?php
$x = 50;
$y = 50;

var_dump($x >= $y); // returns true because $x is greater than or equal to $y
?>

</body>
</html>
```

Result:

bool(true)

Comparison operator: Less than or equal (<=)

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

<?php
$x = 50;
$y = 50;

var_dump($x <= $y); // returns true because $x is less than or equal to $y
?>

</body>
</html>
```

Result:

bool(true)

Increment operator: ++$x

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

<?php
$x = 10;
echo ++$x;
?>

</body>
</html>
```

Result:

11

Increment operator: $x++

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

<?php
$x = 10;
echo $x++;
?>

</body>
</html>
```

Result:

10

Decrement operator: --$x

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

```
<body>

<?php
$x = 10;
echo --$x;
?>

</body>
</html>
```

Result:

9

Decrement operator: $x—

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

<?php
$x = 10;
echo $x--;
?>

</body>
</html>
```

Result:

10

Logical operator: and

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

<?php
$x = 100;
$y = 50;
```

```
if ($x == 100 and $y == 50) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

Logical operator: or

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

<?php
$x = 100;
$y = 50;

if ($x == 100 or $y == 80) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

Logical operator: xor

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

```
<?php
$x = 100;
$y = 50;

if ($x == 100 xor $y == 80) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

Logical operator: && (and)

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

<?php
$x = 100;
$y = 50;

if ($x == 100 && $y == 50) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

Logical operator: || (or)

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

<?php
$x = 100;
$y = 50;

if ($x == 100 || $y == 80) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

Logical operator: not

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

<?php
$x = 100;

if ($x !== 90) {
    echo "Hello world!";
}
?>

</body>
</html>
```

Result:

Hello world!

String operator: Concatenation of $txt1 and $txt2

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

<?php
$txt1 = "Hello";
$txt2 = " world!";
echo $txt1 . $txt2;
?>

</body>
</html>
```

Result:

Hello world!

String operator: Appends $txt2 to $txt1

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

<?php
$txt1 = "Hello";
$txt2 = " world!";
echo $txt1 .= $txt2;
?>

</body>
</html>
```

Result:

Hello world!

Array operator: Union (+)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

print_r($x + $y); // union of $x and $y
?>

</body>
</html>
```

Result:

Array ([a] => red [b] => green [c] => blue [d] => yellow)

Array operator: Equality (==)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

var_dump($x == $y);
?>

</body>
</html>
```

Result:

bool(false)

Array operator: Identity (===)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

var_dump($x === $y);
?>

</body>
</html>
```

Result:

bool(false)

Array operator: Inequality (!=)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

var_dump($x != $y);
?>

</body>
</html>
```

Result:

bool(true)

Array operator: Inequality (<>)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

var_dump($x <> $y);
?>

</body>
</html>
```

Result:

bool(true)

Array operator: Non-identity (!==)

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

<?php
$x = array("a" => "red", "b" => "green");
$y = array("c" => "blue", "d" => "yellow");

var_dump($x !== $y);
?>

</body>
</html>
```

Result:

bool(true)

PHP If...Else and Switch Statements

The if statement

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

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

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

</body>
</html>
```

Result:

The if...else statement

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

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

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

</body>
</html>
```

Result:

Have a good night!

The 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>
```

Result:

The hour (of the server) is 20, and will give the following message:

Have a good night!

The 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, or green!";
}
?>

</body>
</html>
```

Result:

Your favorite color is red!

PHP While and For Loops

The while loop

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

<?php
$x = 1;

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

</body>
</html>
```

Result:

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

The do...while loop

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

<?php
$x = 1;

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

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

</body>
</html>
```

Result:

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

Another do...while loop

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

<?php
$x = 6;

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

</body>
</html>
```

Result:

The number is: 6

The for loop

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

```
<body>

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

</body>
</html>
```

Result:

The number is: 0
The number is: 1
The number is: 2
The number is: 3
The number is: 4
The number is: 5
The number is: 6
The number is: 7
The number is: 8
The number is: 9
The number is: 10

PHP Functions

Create a function

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

<?php
function writeMsg() {
    echo "Hello world!";
}

writeMsg();
?>

</body>
</html>
```

Result:

Hello world!

Function with one argument

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

<?php
function familyName($fname) {
    echo "$fname Refsnes.<br>";
}

familyName("Jani");
familyName("Hege");
familyName("Stale");
familyName("Kai Jim");
familyName("Borge");
?>
```

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

Result:

Jani Refsnes.
Hege Refsnes.
Stale Refsnes.
Kai Jim Refsnes.
Borge Refsnes.

Function with two arguments

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

<?php
function familyName($fname, $year) {
    echo "$fname Refsnes. Born in $year <br>";
}

familyName("Hege","1975");
familyName("Stale","1978");
familyName("Kai Jim","1983");
?>

</body>
</html>
```

Result:

Hege Refsnes. Born in 1975
Stale Refsnes. Born in 1978
Kai Jim Refsnes. Born in 1983

Function with default argument value

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

<?php
function setHeight($minheight = 50) {
    echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight();
setHeight(135);
setHeight(80);
?>

</body>
</html>
```

Result:

The height is : 350
The height is : 50
The height is : 135
The height is : 80

Function that returns a value

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

<?php
function sum($x, $y) {
    $z = $x + $y;
    return $z;
}

echo "5 + 10 = " . sum(5,10) . "<br>";
echo "7 + 13 = " . sum(7,13) . "<br>";
echo "2 + 4 = " . sum(2,4);
?>
```

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

Result:

5 + 10 = 15
7 + 13 = 20
2 + 4 = 6

PHP Arrays

Indexed arrays

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

<?php
$cars = array("Volvo", "BMW", "Toyota");
echo "I like " . $cars[0] . ", " . $cars[1] . " and " . $cars[2] . ".";
?>

</body>
</html>
```

Result:

I like Volvo, BMW and Toyota.

count() - Return the length of an array

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

<?php
$cars = array("Volvo", "BMW", "Toyota");
echo count($cars);
?>

</body>
</html>
```

Result:

3

Loop through an indexed array

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

<?php
$cars = array("Volvo", "BMW", "Toyota");
$arrlength = count($cars);

for($x = 0; $x <  $arrlength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Volvo
BMW
Toyota

Associative arrays

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

<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>

</body>
</html>
```

Result:

Peter is 35 years old.

Loop through an associative array

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

<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Key=Peter, Value=35
Key=Ben, Value=37
Key=Joe, Value=43

PHP Sorting Arrays

sort() - Sort array in ascending alphabetical order

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

<?php
$cars = array("Volvo", "BMW", "Toyota");
sort($cars);

$clength = count($cars);
for($x = 0; $x <  $clength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

BMW
Toyota
Volvo

sort() - Sort array in ascending numerical order

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

<?php
$numbers = array(4, 6, 2, 22, 11);
sort($numbers);

$arrlength = count($numbers);
for($x = 0; $x <  $arrlength; $x++) {
```

```
    echo $numbers[$x];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

2
4
6
11
22

rsort() - Sort array in descending alphabetical order

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

<?php
$cars = array("Volvo", "BMW", "Toyota");
rsort($cars);

$clength = count($cars);
for($x = 0; $x <  $clength; $x++) {
    echo $cars[$x];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Volvo
Toyota
BMW

rsort() - Sort array in descending numerical order

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

<?php
$numbers = array(4, 6, 2, 22, 11);
rsort($numbers);

$arrlength = count($numbers);
for($x = 0; $x <  $arrlength; $x++) {
    echo $numbers[$x];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

22
11
6
4
2

asort() - Sort array in ascending order, according to value

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

<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
asort($age);

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>
```

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

Result:

Key=Peter, Value=35
Key=Ben, Value=37
Key=Joe, Value=43

ksort() - Sort array in ascending order, according to key

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

<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
ksort($age);

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Key=Ben, Value=37
Key=Joe, Value=43
Key=Peter, Value=35

arsort() - Sort array in descending order, according to value

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

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
arsort($age);

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Key=Joe, Value=43
Key=Ben, Value=37
Key=Peter, Value=35

krsort() - Sort array in descending order, according to key

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

<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
krsort($age);

foreach($age as $x => $x_value) {
    echo "Key=" . $x . ", Value=" . $x_value;
    echo "<br>";
}
?>

</body>
</html>
```

Result:

Key=Peter, Value=35
Key=Joe, Value=43
Key=Ben, Value=37

PHP Superglobals

$GLOBAL - Used to access global variables from anywhere in the PHP script

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

<?php
$x = 75;
$y = 25;

function addition() {
    $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
}

addition();
echo $z;
?>

</body>
</html>
```

Result:

100

$_SERVER - Holds information about headers, paths, and script locations

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

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

```
echo "<br>";
echo $_SERVER['HTTP_REFERER'];
echo "<br>";
echo $_SERVER['HTTP_USER_AGENT'];
echo "<br>";
echo $_SERVER['SCRIPT_NAME'];
?>

</body>
</html>
```

Result:

/php/demo_global_server.php
www.w3schools.com
www.w3schools.com
http://www.w3schools.com/php/showphp.asp?filename=demo_global_server
Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.90 Safari/537.36
/php/demo_global_server.php

$_REQUEST - Used to collect data after submitting an HTML form

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

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_REQUEST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
```

```
?>

</body>
</html>
```

Result:

Name: Submit

$_POST - Used to collect form data after submitting an HTML form. Also used to pass variables

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

<form method="post" action="<?php echo $_SERVER['PHP_SELF'];?>">
  Name: <input type="text" name="fname">
  <input type="submit">
</form>

<?php
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    // collect value of input field
    $name = $_POST['fname'];
    if (empty($name)) {
        echo "Name is empty";
    } else {
        echo $name;
    }
}
?>

</body>
</html>
```

Result:

Name: Submit

$_GET - Collect data sent in the URL

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

<a href="test_get.php?subject=PHP&web=W3schools.com">Test $GET</a>

</body>
</html>
```

Test $GET

PHP Form Validation

PHP Form Validation

```
<!DOCTYPE HTML>
<html>
<head>
<style>
.error {color: #FF0000;}
</style>
</head>
<body>

<?php
// define variables and set to empty values
$nameErr = $emailErr = $genderErr = $websiteErr = "";
$name = $email = $gender = $comment = $website = "";

if ($_SERVER["REQUEST_METHOD"] == "POST") {
  if (empty($_POST["name"])) {
    $nameErr = "Name is required";
  } else {
    $name = test_input($_POST["name"]);
    // check if name only contains letters and whitespace
    if (!preg_match("/^[a-zA-Z ]*$/",$name)) {
      $nameErr = "Only letters and white space allowed";
    }
  }

  if (empty($_POST["email"])) {
    $emailErr = "Email is required";
  } else {
    $email = test_input($_POST["email"]);
    // check if e-mail address is well-formed
    if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {
      $emailErr = "Invalid email format";
    }
  }

  if (empty($_POST["website"])) {
    $website = "";
  } else {
    $website = test_input($_POST["website"]);
```

```
    // check if URL address syntax is valid (this regular expression also allows dashes in the URL)
    if (!preg_match("/\b(?:(?:https?|ftp):\/\/|www\.)[-a-z0-9+&@#\/%?=~_|!:,.;]*[-a-z0-
9+&@#\/%=~_|]/i",$website)) {
      $websiteErr = "Invalid URL";
    }
  }

  if (empty($_POST["comment"])) {
    $comment = "";
  } else {
    $comment = test_input($_POST["comment"]);
  }

  if (empty($_POST["gender"])) {
    $genderErr = "Gender is required";
  } else {
    $gender = test_input($_POST["gender"]);
  }
}

function test_input($data) {
  $data = trim($data);
  $data = stripslashes($data);
  $data = htmlspecialchars($data);
  return $data;
}
?>

<h2>PHP Form Validation Example</h2>
<p><span class="error">* required field.</span></p>
<form method="post" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]);?>">
  Name: <input type="text" name="name" value="<?php echo $name;?>">
  <span class="error">* <?php echo $nameErr;?></span>
  <br><br>
  E-mail: <input type="text" name="email" value="<?php echo $email;?>">
  <span class="error">* <?php echo $emailErr;?></span>
  <br><br>
  Website: <input type="text" name="website" value="<?php echo $website;?>">
  <span class="error"><?php echo $websiteErr;?></span>
  <br><br>
  Comment: <textarea name="comment" rows="5" cols="40"><?php echo $comment;?></textarea>
  <br><br>
  Gender:
  <input type="radio" name="gender" <?php if (isset($gender) && $gender=="female") echo
"checked";?>  value="female">Female
  <input type="radio" name="gender" <?php if (isset($gender) && $gender=="male") echo
```

```
"checked";?>  value="male">Male
  <span class="error">* <?php echo $genderErr;?></span>
  <br><br>
  <input type="submit" name="submit" value="Submit">
</form>

<?php
echo "<h2>Your Input:</h2>";
echo $name;
echo "<br>";
echo $email;
echo "<br>";
echo $website;
echo "<br>";
echo $comment;
echo "<br>";
echo $gender;
?>

</body>
</html>
```

Result:

PHP Form Validation Example

* required field.

Name: *

E-mail: *

Website:

Comment:

Gender: Female Male *

Submit

Your Input:

PHP Multidimensional Arrays

Output elements from a multidimensional array

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

<?php
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
  );

echo $cars[0][0].": In stock: ".$cars[0][1].", sold: ".$cars[0][2].".<br>";
echo $cars[1][0].": In stock: ".$cars[1][1].", sold: ".$cars[1][2].".<br>";
echo $cars[2][0].": In stock: ".$cars[2][1].", sold: ".$cars[2][2].".<br>";
echo $cars[3][0].": In stock: ".$cars[3][1].", sold: ".$cars[3][2].".<br>";
?>

</body>
</html>
```

Result:

Volvo: In stock: 22, sold: 18.
BMW: In stock: 15, sold: 13.
Saab: In stock: 5, sold: 2.
Land Rover: In stock: 17, sold: 15.

Loop through a multidimensional array

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

<?php
```

```
$cars = array
  (
  array("Volvo",22,18),
  array("BMW",15,13),
  array("Saab",5,2),
  array("Land Rover",17,15)
  );

for ($row = 0; $row <  4; $row++) {
  echo "<p><b>Row number $row</b></p>";
  echo "<ul>";
  for ($col = 0; $col <  3; $col++) {
    echo "<li>".$cars[$row][$col]."</li>";
  }
  echo "</ul>";
}
?>

</body>
</html>
```

Result:

Row number 0

- Volvo
- 22
- 18

Row number 1

- BMW
- 15
- 13

Row number 2

- Saab
- 5
- 2

Row number 3

- Land Rover
- 17
- 15

PHP Date and Time

Format today's date in several ways

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

<?php
echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
echo "Today is " . date("Y-m-d") . "<br>";
echo "Today is " . date("l");
?>

</body>
</html>
```

Result:

Today is 2015/04/28
Today is 2015.04.28
Today is 2015-04-28
Today is Tuesday

Automatically update the copyright year on your website

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

© 2010-<?php echo date("Y")?>

</body>
</html>
```

Result:

© 2010-2015

Output the current time (server time)

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

<?php
echo "The time is " . date("h:i:sa");
?>

</body>
</html>
```

Result:

The time is 08:24:09pm

Set timezone, then output current time

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

<?php
date_default_timezone_set("America/New_York");
echo "The time is " . date("h:i:sa");
?>

</body>
</html>
```

Result:

The time is 08:24:35pm

Create a date and time from a number of parameters in mktime()

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

```
<?php
$d=mktime(11, 14, 54, 8, 12, 2014);
echo "Created date is " . date("Y-m-d h:i:sa", $d);
?>

</body>
</html>
```

Result:

Created date is 2014-08-12 11:14:54am

Create a date and time from the strtotime() function

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

<?php
$d=strtotime("10:30pm April 15 2014");
echo "Created date is " . date("Y-m-d h:i:sa", $d);
?>

</body>
</html>
```

Result:

Created date is 2014-04-15 10:30:00pm

Create more dates/times from strtotime()

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

<?php
$d=strtotime("tomorrow");
echo date("Y-m-d h:i:sa", $d) . "<br>";
```

```
$d=strtotime("next Saturday");
echo date("Y-m-d h:i:sa", $d) . "<br>";

$d=strtotime("+3 Months");
echo date("Y-m-d h:i:sa", $d) . "<br>";
?>

</body>
</html>
```

Result:

2015-04-29 12:00:00am
2015-05-02 12:00:00am
2015-07-28 08:25:47pm

Output the dates for the next six Saturdays

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

<?php
$startdate=strtotime("Saturday");
$enddate=strtotime("+6 weeks",$startdate);

while ($startdate <  $enddate) {
  echo date("M d", $startdate),"<br>";
  $startdate = strtotime("+1 week", $startdate);
}
?>

</body>
</html>
```

Result:

May 02
May 09
May 16
May 23

May 30
Jun 06

Output the number of days until 4th of July

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

<?php
$d1=strtotime("July 04");
$d2=ceil(($d1-time())/60/60/24);
echo "There are " . $d2 ." days until 4th of July.";
?>

</body>
</html>
```

Result:

There are 67 days until 4th of July.

PHP Include Files

Use include to include "footer.php" in a page

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

<h1>Welcome to my home page!</h1>
<p>Some text.</p>
<p>Some more text.</p>
<?php include 'footer.php';?>

</body>
</html>
```

Result:

Welcome to my home page!

Some text.

Some more text.

Copyright © 1999-2015 W3Schools.com

Use include to include "menu.php" in a page

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

<div class="menu">
<?php include 'menu.php';?>
</div>

<h1>Welcome to my home page!</h1>
```

```
<p>Some text.</p>
<p>Some more text.</p>

</body>
</html>
```

Result:

Home - HTML Tutorial - CSS Tutorial - JavaScript Tutorial - PHP Tutorial

Welcome to my home page!

Some text.

Some more text.

Use include to include "vars.php" in a page

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

<h1>Welcome to my home page!</h1>
<?php include 'vars.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

Result:

Welcome to my home page!

I have a red BMW.

Use include to include a non-existing file

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

<h1>Welcome to my home page!</h1>
<?php include 'noFileExists.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

Result:

Welcome to my home page!

I have a .

Use require to include a non-existing file

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

<h1>Welcome to my home page!</h1>
<?php require 'noFileExists.php';
echo "I have a $color $car.";
?>

</body>
</html>
```

Result:

Welcome to my home page!

PHP File Handling

Use readfile() to read a file and write it to the output buffer

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

<?php
echo readfile("webdictionary.txt");
?>

</body>
</html>
```

Result:

AJAX = Asynchronous JavaScript and XML CSS = Cascading Style Sheets HTML = Hyper Text Markup Language PHP = PHP Hypertext Preprocessor SQL = Structured Query Language SVG = Scalable Vector Graphics XML = EXtensible Markup Language236

PHP File Open/Read/Close

Use fopen(), fread(), and fclose() to open, read, and close a file

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

<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
echo fread($myfile,filesize("webdictionary.txt"));
fclose($myfile);
?>

</body>
</html>
```

Result:

AJAX = Asynchronous JavaScript and XML CSS = Cascading Style Sheets HTML = Hyper Text Markup Language PHP = PHP Hypertext Preprocessor SQL = Structured Query Language SVG = Scalable Vector Graphics XML = EXtensible Markup Language

Use fgets() to read a single line from a file

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

<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
echo fgets($myfile);
fclose($myfile);
?>

</body>
</html>
```

Result:

AJAX = Asynchronous JavaScript and XML

Use feof() to read through a file, line by line, until end-of-file is reached

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

<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
// Output one line until end-of-file
while(!feof($myfile)) {
  echo fgets($myfile) . "<br>";
}
fclose($myfile);
?>

</body>
</html>
```

Result:

AJAX = Asynchronous JavaScript and XML
CSS = Cascading Style Sheets
HTML = Hyper Text Markup Language
PHP = PHP Hypertext Preprocessor
SQL = Structured Query Language
SVG = Scalable Vector Graphics
XML = EXtensible Markup Language

Use fgetc() to read a single character from a file

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

<?php
$myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!");
// Output one character until end-of-file
```

```
while(!feof($myfile)) {
  echo fgetc($myfile);
}
fclose($myfile);
?>

</body>
</html>
```

Result:

AJAX = Asynchronous JavaScript and XML CSS = Cascading Style Sheets HTML = Hyper Text Markup Language PHP = PHP Hypertext Preprocessor SQL = Structured Query Language SVG = Scalable Vector Graphics XML = EXtensible Markup Language

PHP Cookies

Create and retrieve a cookie

```
<!DOCTYPE html>
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

<p><strong>Note:</strong> You might have to relaod the page to see the value of the cookie.</p>

</body>
</html>
```

Result:

Cookie named 'user' is not set!

Note: You might have to relaod the page to see the value of the cookie.

Modify a cookie value

```
<!DOCTYPE html>
<?php
$cookie_name = "user";
$cookie_value = "Alex Porter";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/");
```

```
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

<p><strong>Note:</strong> You might have to relaod the page to see the new value of the cookie.</p>

</body>
</html>
```

Result:

Cookie 'user' is set!
Value is: John Doe

Note: You might have to relaod the page to see the new value of the cookie.

Delete a cookie

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

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

</body>
</html>
```

Result:

Cookie 'user' is deleted.

Check if cookies are enabled

```
<!DOCTYPE html>
<?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>
```

Result:

Cookies are enabled.

PHP Sessions

Start a session

```
<?php
// Start the session
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Set session variables
$_SESSION["favcolor"] = "green";
$_SESSION["favanimal"] = "cat";
echo "Session variables are set.";
?>

</body>
</html>
```

Result:

Session variables are set.

Get session variable values

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// Echo session variables that were set on previous page
echo "Favorite color is " . $_SESSION["favcolor"] . ".<br>";
echo "Favorite animal is " . $_SESSION["favanimal"] . ".";
?>
```

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

Result:

Favorite color is green.
Favorite animal is cat.

Get all session variable values

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
print_r($_SESSION);
?>

</body>
</html>
```

Result:

Array ([favcolor] => green [favanimal] => cat)

Modify a session variable

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// to change a session variable, just overwrite it
```

```
$_SESSION["favcolor"] = "yellow";
print_r($_SESSION);
?>

</body>
</html>
```

Result:

Array ([favcolor] => yellow [favanimal] => cat)

Destroy a session

```
<?php
session_start();
?>
<!DOCTYPE html>
<html>
<body>

<?php
// remove all session variables
session_unset();

// destroy the session
session_destroy();

echo "All session variables are now removed, and the session is destroyed."
?>

</body>
</html>
```

Result:

All session variables are now removed, and the session is destroyed.

PHP Filters

Use filter_list() to list what the PHP filter extension offers

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
th, td {
    padding: 5px;
}
</style>
</head>
<body>

<table>
  <tr>
    <td>Filter Name</td>
    <td>Filter ID</td>
  </tr>
  <?php
  foreach (filter_list() as $id =>$filter) {
     echo '<tr><td>' . $filter . '</td><td>' . filter_id($filter) . '</td></tr>';
  }
  ?>
</table>

</body>
</html>
```

Result:

Filter Name	Filter ID
int	257
boolean	258
float	259
validate_regexp	272

validate_url	273
validate_email	274
validate_ip	275
string	513
stripped	513
encoded	514
special_chars	515
full_special_chars	522
unsafe_raw	516
email	517
url	518
number_int	519
number_float	520
magic_quotes	521
callback	1024

Sanitize a string

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

<?php
$str = "<h1>Hello World!</h1>";
$newstr = filter_var($str, FILTER_SANITIZE_STRING);
echo $newstr;
?>

</body>
</html>
```

Result:

Hello World!

Validate an integer

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

<?php
$int = 100;

if (!filter_var($int, FILTER_VALIDATE_INT) === false) {
    echo("Integer is valid");
} else {
    echo("Integer is not valid");
}
?>

</body>
</html>
```

Result:

Integer is valid

Validate an integer that is 0

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

<?php
$int = 0;

if (filter_var($int, FILTER_VALIDATE_INT) === 0 || !filter_var($int, FILTER_VALIDATE_INT)
=== false) {
    echo("Integer is valid");
} else {
    echo("Integer is not valid");
}
?>

</body>
</html>
```

Result:

Integer is valid

Validate an IP address

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

<?php
$ip = "127.0.0.1";

if (!filter_var($ip, FILTER_VALIDATE_IP) === false) {
    echo("$ip is a valid IP address");
} else {
    echo("$ip is not a valid IP address");
}
?>

</body>
</html>
```

Result:

127.0.0.1 is a valid IP address

Sanitize and validate an email address

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

<?php
$email = "john.doe@example.com";

// Remove all illegal characters from email
$email = filter_var($email, FILTER_SANITIZE_EMAIL);

// Validate e-mail
if (!filter_var($email, FILTER_VALIDATE_EMAIL) === false) {
    echo("$email is a valid email address");
```

```
} else {
    echo("$email is not a valid email address");
}
?>

</body>
</html>
```

Result:

john.doe@example.com is a valid email address

Sanitize and validate a URL

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

<?php
$url = "http://www.w3schools.com";

// Remove all illegal characters from a url
$url = filter_var($url, FILTER_SANITIZE_URL);

// Validate url
if (!filter_var($url, FILTER_VALIDATE_URL) === false) {
    echo("$url is a valid URL");
} else {
    echo("$url is not a valid URL");
}
?>

</body>
</html>
```

Result:

http://www.w3schools.com is a valid URL

PHP Select Data From MySQL

Select data with MySQLi (Object-oriented)

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

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<br> id: ". $row["id"]. " - Name: ". $row["firstname"]. " " . $row["lastname"] . "<br>";
    }
} else {
    echo "0 results";
}

$conn->close();
?>

</body>
</html>
```

Result:

id: 1 - Name: John Doe
id: 2 - Name: Mary Moe
id: 3 - Name: Julie Dooley

Select data with MySQLi (Object-oriented) and put result in an HTML table

```
<!DOCTYPE html>
<html>
<head>
<style>
table, th, td {
    border: 1px solid black;
}
</style>
</head>
<body>

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    echo "<table><tr><th>ID</th><th>Name</th></tr>";
    // output data of each row
    while($row = $result->fetch_assoc()) {
        echo "<tr><td>" . $row["id"]. "</td><td>" . $row["firstname"]. " " . $row["lastname"].
"</td></tr>";
    }
    echo "</table>";
```

```
} else {
    echo "0 results";
}

$conn->close();
?>

</body>
</html>
```

Result:

ID	Name
1	John Doe
2	Mary Moe
3	Julie Dooley

Select data with MySQLi (Procedural)

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

<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
   die("Connection failed: " . mysqli_connect_error());
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = mysqli_query($conn, $sql);

if (mysqli_num_rows($result) > 0) {
   // output data of each row
   while($row = mysqli_fetch_assoc($result)) {
      echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
```

```
    }
} else {
    echo "0 results";
}

mysqli_close($conn);
?>

</body>
</html>
```

Result:

id: 1 - Name: John Doe
id: 2 - Name: Mary Moe
id: 3 - Name: Julie Dooley

Select data with PDO (+ Prepared statements)

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

<?php
echo "<table style='border: solid 1px black;'>";
  echo "<tr><th>Id</th><th>Firstname</th><th>Lastname</th></tr>";

class TableRows extends RecursiveIteratorIterator {
    function __construct($it) {
        parent::__construct($it, self::LEAVES_ONLY);
    }

    function current() {
        return "<td style='width: 150px; border: 1px solid black;'>" . parent::current(). "</td>";
    }

    function beginChildren() {
        echo "<tr>";
    }

    function endChildren() {
        echo "</tr>" . "\n";
    }
```

```
}

$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDBPDO";

try {
    $conn = new PDO("mysql:host=$servername;dbname=$dbname", $username, $password);
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $stmt = $conn->prepare("SELECT id, firstname, lastname FROM MyGuests");
    $stmt->execute();

    // set the resulting array to associative
    $result = $stmt->setFetchMode(PDO::FETCH_ASSOC);

    foreach(new TableRows(new RecursiveArrayIterator($stmt->fetchAll())) as $k=>$v) {
        echo $v;
    }
}
catch(PDOException $e) {
    echo "Error: " . $e->getMessage();
}
$conn = null;
echo "</table>";
?>

</body>
</html>
```

Result:

Id	Firstname	Lastname
1	John	Doe
2	Mary	Moe
3	Julie	Dooley

PHP SimpleXML Parser

Use simplexml_load_string() to read XML data from a string

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

<?php
$myXMLData =
"<?xml version='1.0' encoding='UTF-8'?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>";

$xml=simplexml_load_string($myXMLData) or die("Error: Cannot create object");
print_r($xml);
?>

</body>
</html>
```

Result:

SimpleXMLElement Object ([to] => Tove [from] => Jani [heading] => Reminder [body] => Don't forget me this weekend!)

Use simplexml_load_file() to read XML data from a file

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

<?php
$xml=simplexml_load_file("note.xml") or die("Error: Cannot create object");
print_r($xml);
?>
```

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

Result:

SimpleXMLElement Object ([to] => Tove [from] => Jani [heading] => Reminder [body] => Don't forget me this weekend!)

Get node values

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

<?php
$xml=simplexml_load_file("note.xml") or die("Error: Cannot create object");
echo $xml->to . "<br>";
echo $xml->from . "<br>";
echo $xml->heading . "<br>";
echo $xml->body;
?>

</body>
</html>
```

Result:

Tove
Jani
Reminder
Don't forget me this weekend!

Get node values of specific elements

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

<?php
```

```
$xml=simplexml_load_file("books.xml") or die("Error: Cannot create object");
echo $xml->book[0]->title . "<br>";
echo $xml->book[1]->title;
?>

</body>
</html>
```

Result:

Everyday Italian
Harry Potter

Get node values - loop

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

<?php
$xml=simplexml_load_file("books.xml") or die("Error: Cannot create object");
foreach($xml->children() as $books) {
        echo $books->title . ", ";
        echo $books->author . ", ";
        echo $books->year . ", ";
        echo $books->price . "<br>";
}
?>

</body>
</html>
```

Result:

Everyday Italian, Giada De Laurentiis, 2005, 30.00
Harry Potter, J K. Rowling, 2005, 29.99
XQuery Kick Start, James McGovern, 2003, 49.99
Learning XML, Erik T. Ray, 2003, 39.95

Get attribute values

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

<?php
$xml=simplexml_load_file("books.xml") or die("Error: Cannot create object");
echo $xml->book[0]['category'] . "<br>";
echo $xml->book[1]->title['lang'];
?>

</body>
</html>
```

Result:

COOKING
en

Get attribute values - loop

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

<?php
$xml=simplexml_load_file("books.xml") or die("Error: Cannot create object");
foreach($xml->children() as $books) {
    echo $books->title['lang'];
    echo "<br>";
}
?>

</body>
</html>
```

Result:

en
en
en-us
en-us

PHP XML Expat Parser

Initialize an XML Expat parser, define some handlers, then parse an XML file

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

<?php
//Initialize the XML parser
$parser=xml_parser_create();

//Function to use at the start of an element
function start($parser,$element_name,$element_attrs) {
    switch($element_name) {
        case "NOTE":
        echo "-- Note --<br>";
    break;
        case "TO":
        echo "To: ";
    break;
        case "FROM":
        echo "From: ";
    break;
        case "HEADING":
        echo "Heading: ";
    break;
        case "BODY":
        echo "Message: ";
    }
}

//Function to use at the end of an element
function stop($parser,$element_name) {
    echo "<br>";
}

//Function to use when finding character data
function char($parser,$data) {
    echo $data;
}
```

```
//Specify element handler
xml_set_element_handler($parser,"start","stop");

//Specify data handler
xml_set_character_data_handler($parser,"char");

//Open XML file
$fp=fopen("note.xml","r");

//Read data
while ($data=fread($fp,4096)) {
    xml_parse($parser,$data,feof($fp)) or
    die (sprintf("XML Error: %s at line %d",
    xml_error_string(xml_get_error_code($parser)),
    xml_get_current_line_number($parser)));
}

//Free the XML parser
xml_parser_free($parser);
?>

</body>
</html>
```

Result:

-- Note --
To: Tove
From: Jani
Heading: Reminder
Message: Don't forget me this weekend!