**PHP Introduction**

## 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
* PHP is free to download and use

**PHP is an amazing and popular language!**

It is powerful enough to be at the core of the biggest blogging system on the web (WordPress)!  
It is deep enough to run the largest social network (Facebook)!  
It is also easy enough to be a beginner's first server-side language!

## What is a PHP File?

* PHP files can contain text, HTML, CSS, JavaScript, and PHP code
* PHP code is executed on the server, and the result is returned to the browser as plain HTML
* PHP files have extension ".php"

## What Can PHP Do?

* PHP can generate dynamic page content
* PHP can create, open, read, write, delete, and close files on the server
* PHP can collect form data
* PHP can send and receive cookies
* PHP can add, delete, modify data in your database
* PHP can be used to control user-access
* PHP can encrypt data

## 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](http://www.php.net/)
* PHP is easy to learn and runs efficiently on the server side

## What's new in PHP 7

* PHP 7 is much faster than the previous popular stable release (PHP 5.6)
* PHP 7 has improved Error Handling
* PHP 7 supports stricter Type Declarations for function arguments
* PHP 7 supports new operators (like the spaceship operator: <=> )

**Basic PHP Syntax**

A PHP script can be placed anywhere in the document.

A PHP script starts with <?php and ends with ?>:

<?php  
// PHP code goes here  
?>

The default file extension for PHP files is ".php".

PHP Case Sensitivity

**In PHP, NO keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are case-sensitive.**

### Example

<!DOCTYPE html>  
<html>  
<body>  
**<?php  
ECHO "Hello World!<br>";  
echo "Hello World!<br>";  
EcHo "Hello World!<br>";  
?>**  
</body>  
</html>

$color, $COLOR, and $coLOR are treated as three different variables:

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

Comments in PHP

**<?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  
\*/**

**?>**

**PHP Variables**

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total\_volume).

Rules for 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 cannot start with a number
* A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_ )
* Variable names are case-sensitive ($age and $AGE are two different variables)

Note All Variables are case sensitive:

**The PHP echo statement is often used to output data to the screen.**

**<?php  
$txt = "W3Schools.com";  
echo "I love $txt!";  
?>**

**<?php  
$txt = "W3Schools.com";  
echo "I love " . $txt . "!";  
?>**

PHP Variables Scope

**PHP has three different variable scopes:**

* Local (inside a function)
* Global (0utside a function)
* Static
* **Variable Declared By (global keyword & super global [$\_GLOBALS])**

Variable with global scope:

<?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>";  
?>

Variable with local scope:

<?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>";  
?>

**PHP The global Keyword**

The global keyword is used to access a global variable from within a function.

<?php  
$x = 5;  
$y = 10;  
  
function myTest() {   global $x, $y;  
     $y = $x + $y;}  
  
myTest();  
echo $y; // outputs 15  
?>

**PHP ($GLOBALS) Superglobals**

<?php  
$x = 5; $y = 10;  
function myTest() {  
    $GLOBALS['y'] = $GLOBALS['x'] + $GLOBALS['y'];}  
myTest();  
echo $y; // outputs 15  
?>

**PHP STATIC Keyword**

**// Let a local variable not to be deleted after the execution of function…**function myTest() {  
    static $x = 0;  
    echo $x;     $x++; }  
myTest();  
myTest();  
myTest();  
?>

PHP Data Types

Variables can store data of different types, and different data types can do different things.

* **String**
* **Integer**
* **Float (floating point numbers - also called double)**
* **Boolean**
* **Array**
* **Object**
* **NULL**
* **Resource**

PHP String

**A string is a sequence of characters, like "Hello world!".**

$x = "Hello world!";  
$y = 'Hello world!';  
echo $x;

PHP Integer

**An integer data type is a non-decimal number between -2,147,483,648 and 2,147,483,647.**

* An integer must have at least one digit
* An integer must not have a decimal point
* An integer can be either positive or negative

**The PHP var\_dump() function returns the data type and value:**

**<?php  
$x = 5985;  
var\_dump($x);  
?>**

PHP Float

**A float (floating point number) is a number with a decimal point**

<?php  
$x = 10.365;  
var\_dump($x);  
?>

PHP Boolean

**A Boolean represents two possible states: TRUE or FALSE.**

$x = true;  
$y = false;

\

PHP Array

**An array stores multiple values in one single variable.**

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

PHP Object

An object is a data type which stores data and information on how to process that data. **First we must declare a class of object**. For this, we use the class keyword. **A class is a structure that can contain properties and methods**:

<?php

class fruits{

var $name = 'apple';//Class property $ value;

var $color = 'RED';

function method(){//Method Declared By Function;

$this->name = 'Orange';//method Property & Value;

$this->price = '$10';

}}

$object = new fruits;//call the main class

$object ->method();//call the method of a function....

echo $object ->name;//properties of the method....

echo $object ->price;//properties of the method.

echo $object ->name."<br>";//Properties of the method....

echo $object ->color;//Properties of the class bez no properties was not declared

PHP NULL Value

Null is a special data type which can have only one value: NULL**. If a variable is created without a value, it is automatically assigned a value of NULL.**

**$x = "Hello world!"; $x = null;  
var\_dump($x);**

PHP Strings

strlen() – (String Length)

echo strlen("Hello world!"); // outputs 12

str\_word\_count() - Count Words in a String

echo str\_word\_count("Hello world!"); // outputs 2

strrev() - Reverse a String

echo strrev("Hello world!"); // outputs !dlrow olleH

strpos() – String Position

echo strpos("Hello world!", "world"); // outputs 6

str\_replace() - Replace Text Within a String

echo str\_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!

**PHP Numbers**

**Check if the type of a variable is integer:**

$x = 5985; var\_dump(is\_int($x));  
  
$x = 59.85; var\_dump(is\_int($x));

**Check if the type of a variable is float:**

$x = 10.365; var\_dump(is\_float($x));

**Check if a numeric value is finite or infinite:**  
 $x = 1.9e411; var\_dump($x);

**Check if the variable is numeric:**

$x = 5985; var\_dump(is\_numeric($x));  
  
$x = "5985"; var\_dump(is\_numeric($x));  
  
$x = "59.85" + 100; var\_dump(is\_numeric($x));  
  
$x = "Hello"; var\_dump(is\_numeric($x));

**Cast float and string to integer:**

|  |  |
| --- | --- |
| **// Cast float to int** $x = 23465.768; $int\_cast = (int)$x; echo $int\_cast; echo "<br>"; | **// Cast string to int** $x = "23465.768"; $int\_cast = (int)$x;  echo $int\_cast; |
|  |  |

**PHP Constant**

## Syntax: define(name, value, case-insensitive)

* First Two Parameter name & value must be in quotes.
* NO need of parameter when you will assign constant.

Constant with case sensitive name:

define("GREETING", "Welcome to W3Schools.com!");  
echo GREETING;

Constant With case in-sensitive name:

define("GREETING", "Welcome to W3Schools.com!", **true);**  
echo greeting;

Create an Array constant:

define("cars", [ "Alfa Romeo",  "BMW",  "Toyota"]);  
echo cars[0];

Constants are Global

define("**GREETING**", "Welcome to W3Schools.com!");  
  
function myTest() {

  echo GREETING;  
}  
 myTest();

**PHP Operators**

**Operators are used to perform operations on variables and values.**

* **Arithmetic operators**
* **Assignment operators (overwrite previous var);**
* **Comparison operators**
* **Increment/Decrement operators**
* **Logical operators**
* **String operators**
* **Array operators**
* **Conditional assignment operators**

PHP 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 | raising $x to the $y'th power |

PHP Assignment Operators

|  |  |  |
| --- | --- | --- |
| **Assignment** | **Same as...** | **Description** |
| x += y | x = x + y | Addition |
| x -= y | x = x - y | Subtraction |
| x \*= y | x = x \* y | Multiplication |

PHP String Operators

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** |
| . | Concate | $txt1 . $txt2 |  |
| .= | Concateassign | $txt1 .= $txt2 | Appends $txt2 to $txt1 |

PHP Logical Operators

|  |  |  |  |
| --- | --- | --- | --- |
| **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 | $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 |

PHP Array Operators

|  |  |  |
| --- | --- | --- |
| **Name** | **Example** | **Result** |
| Union | $x + $y | Union of $x and $y |
| Equality | $x == $y | if $x and $y have the same key/value |
| 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 |
| Non-identity | $x !== $y | Returns true if $x is not identical to $y |

**PHP if...else...elseif Statements**

**[The if statement executes some code if one condition is true.](https://www.w3schools.com/php/php_operators.asp)**

$t = date("H");  
if ($t < "10") {  
    echo "Have a good morning!";  
} elseif ($t < "20") {  
    echo "Have a good day!";  
} else {  
    echo "Have a good night!";  
}

**The PHP switch Statement**

**$favcolor = "red";  
  
switch ($favcolor) {  
    case "red":  
        echo "Your favorite color is red!";  
        break;  
        case "green":  
        echo "Your favorite color is green!";  
        break;  
    default:  
        echo "Your favorite color is neither red, blue, nor green!";**

**}**

**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

The PHP while Loop

|  |  |
| --- | --- |
| $x = 1;    while($x <= 5) {  echo "The number is: $x <br>";  $x++;  } | The number is: 1 The number is: 2 The number is: 3 The number is: 4 The number is: 5 |

PHP do while Loop

|  |  |
| --- | --- |
| $x = 1;  do {  echo "The number is: $x <br>";  $x++;  } while ($x <= 5); | The number is: 1 The number is: 2 The number is: 3 The number is: 4 The number is: 5 |
|  |  |

**Note:** In a do...while loop the condition is tested AFTER executing the statements within the loop. This means that the do...while loop will execute its statements at least once, even if the condition is false. See example below.

|  |  |
| --- | --- |
| $x = 6;  do {     echo "The number is: $x <br>";     $x++; } while ($x <= 5); | The number is: 6 |

The PHP for Loop

|  |  |
| --- | --- |
| <?php for ($x = 0; $x <= 3; $x++) {     echo "The number is: $x <br>"; } ?> | The number is: 0 The number is: 1 The number is: 2 The number is: 3 |

|  |  |
| --- | --- |
| for ($x = 0; $x <= 50; $x+=10) {     echo "The number is: $x <br>"; } | The number is: 0 The number is: 10 The number is: 20 The number is: 30 |

* **$x+=10 - Increase the loop counter value by 10 for each iteration**

# **PHP FOREACH Loop**

|  |  |
| --- | --- |
| $colors = array("red", "green", "blue", "yellow"); foreach ($colors as $value) {   echo "$value <br>";} | red green blue yellow |

# PHP Functions

**Note: A function name must start with a letter or an underscore. Function names are NOT case-sensitive.**

|  |  |
| --- | --- |
| function writeMsg() {     echo "Hello world!"; }  writeMsg(); // call the function | Hello world! |

PHP Function Arguments

|  |  |
| --- | --- |
| function familyName($fname) {     echo "$fname Refsnes.<br>"; }  familyName("Jani"); familyName("Hege"); familyName("Stale"); familyName("Kai Jim"); | Jani Refsnes. Hege Refsnes. Stale Refsnes. Kai Jim Refsnes. Borge Refsnes. |

PHP Functions - Returning values

|  |  |
| --- | --- |
| function sum(int $x, int $y) {     $z = $x + $y;     return $z; } echo "5 + 10 = " . sum(5, 10) . "<br>"; echo "7 + 13 = " . sum(7, 13) . "<br>"; | 5 + 10 = 15 7 + 13 = 20 |

# Create an Array in PHP

In PHP, there are three types of arrays:

* **Indexed arrays** - Arrays with a numeric index
* **Associative arrays** - Arrays with named keys
* **Multidimensional arrays** -Containing one or more arrays

Loop Through an Indexed Array

|  |  |
| --- | --- |
| $cars = array("Volvo", "BMW", "Toyota"); $arrlength = count($cars);  for($x = 0; $x < $arrlength; $x++) {     echo $cars[$x];     echo "<br>"; } | Volvo BMW Toyota |

PHP Associative Arrays

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

PHP Multidimensional Arrays

|  |
| --- |
| <?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>"; |
| 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 Multidimensional Array

|  |  |
| --- | --- |
| <?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>";}  ?> | **Row number 0**   * Volvo * 22 * 18   **Row number 1**   * BMW * 15 * 13   **Row number 2**   * Saab * 5 * 2   **Row number 3**   * Land Rover * 17 |

**PHP - Sort Functions For Arrays**

* sort() - ascending order name & numeric
* rsort() - descending order name & numeric
* asort() - ascending order, according to the value
* ksort() - ascending order, according to the key
* arsort() - descending order, according to the value
* krsort() - descending order, according to the key

|  |  |
| --- | --- |
| Sort Array in Ascending Order - sort() | |
| <?php $cars = array("Volvo", "BMW", "Toyota"); sort($cars); ?> | BMW Toyota Volvo |
| <?php $numbers = array(4, 6, 2, 22, 11); sort($numbers); ?> | 2 4 6 11 22 |

|  |  |
| --- | --- |
| **Sort Array in Descending Order - rsort()** | |
| <?php $cars = array("Volvo", "BMW", "Toyota"); rsort($cars); ?> | Volvo Toyota BMW |

|  |  |
| --- | --- |
| Sort Array (Ascending Order), According to Value - asort() | |
| <?php $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43"); asort($age); ?> | Key=Peter, Value=35 Key=Ben, Value=37 Key=Joe, Value=43 |

|  |  |
| --- | --- |
| Sort Array (Ascending Order), According to Key - ksort() | |
| <?php $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43"); ksort($age); ?> | Key=Ben, Value=37 Key=Joe, Value=43 Key=Peter, Value=35 |

**PHP Global Variables - SUPERGLOBALS**

The PHP SUPERGLOBAL variables are:

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

|  |  |
| --- | --- |
| PHP $GLOBALS | |
| $x = 75; $y = 25;   function addition() {     $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y']; }   addition(); echo $z; | 100 |

|  |  |
| --- | --- |
| **Element/Code** | **Description** |
| $\_SERVER['PHP\_SELF'] | Returns the filename of the currently executing script |
| $\_SERVER['SERVER\_ADDR'] | Returns the IP address of the host server |
| $\_SERVER['SERVER\_NAME'] | Returns the name of the host server (such as www.w3schools.com) |
| $\_SERVER['REQUEST\_METHOD'] | Returns the request method used to access the page (such as POST) |
| $\_SERVER['REQUEST\_TIME'] | Returns the timestamp of the start of the request (such as 1377687496) |
| $\_SERVER['QUERY\_STRING'] | Returns the query string if the page is accessed via a query string |
| $\_SERVER['HTTP\_ACCEPT'] | Returns the Accept header from the current request |
| $\_SERVER['HTTP\_ACCEPT\_CHARSET'] | Returns the Accept\_Charset header from the current request (such as utf-8,ISO-8859-1) |
| $\_SERVER['HTTPS'] | Is the script queried through a secure HTTP protocol |
| $\_SERVER['REMOTE\_ADDR'] | Returns the IP address from where the user is viewing the current page |
| $\_SERVER['REMOTE\_HOST'] | Returns the Host name from where the user is viewing the current page |
| $\_SERVER['SERVER\_PORT'] | Returns the port on the server machine being used by the web server for communication (such as 80) |
| $\_SERVER['PATH\_TRANSLATED'] | Returns the file system-based path to the current script |

PHP **Form Validation**

|  |  |
| --- | --- |
| **Field** | **Validation Rules** |
| Name | Required. + Must only contain letters and whitespace |
| E-mail | Required. + Must contain a valid email address (with @ and .) |
| Website | Optional. If present, it must contain a valid URL |
| Comment | Optional. Multi-line input field (textarea) |
| Gender | Required. Must select one |

The $\_SERVER["PHP\_SELF"] is a super global variable that returns the filename of the currently executing script.

**What is the htmlspecialchars() function?**  
  
The htmlspecialchars() function converts special characters to HTML entities. This means that it will replace HTML characters like < and > with &lt; and &gt;. This prevents attackers from exploiting the code by injecting HTML or Javascript code (Cross-site Scripting attacks) in forms.

**Cross-site scripting (XSS) is a type of computer security vulnerability typically found in Web applications. XSS enables attackers to inject client-side script into Web pages viewed by other users.**

**//From Validation Code in PHP**

# <?php // Flag Variables $nameErr = $emailErr = $genderErr = $websiteErr = ""; $name = $email = $gender = $comment = $website = ""; if ($\_SERVER["REQUEST\_METHOD"] == "POST") {

# //Form Name Validation.......

# if (!empty($\_POST['name'])) {

# if (preg\_match("/^[a-zA-Z ]\*$/", $\_POST['name'])) {

# 

# $name = valid($\_POST['name']);

# }else{

# $name\_err = "Name Is Invalid";

# }

# }else{ $name\_err = "Name Is Requred"; }

# //Form Email Validation......

# if (!empty($\_POST['email'])) {

# if (filter\_var($\_POST['email'],FILTER\_VALIDATE\_EMAIL)) {

# $email = valid($\_POST['email']);

# 

# }else{ $email\_err = "Your Email Is Unvalid";}

# }else{ $email\_err = "Email is Requred";}

# 

# if (!empty($\_POST['web'])) {

# 

# if (filter\_var($\_POST['web'], FILTER\_VALIDATE\_URL)) {

# 

# $web = valid($\_POST['web']);

# }else{

# $web\_err = "Invalid Web URL";

# }

# }else{

# $web\_err = "website name is requred";

# }

# //comment validation.....

# if (!empty($\_POST['comment'])) {

# $comment = valid($\_POST['comment']);

# }

# //Gender validation......

# if (!empty($\_POST['gender'])) {

# 

# $gender = valid($\_POST['gender']);

# }else{ $gender\_err = "Gender is required";}} function test\_input($data) {   $data = trim($data);   $data = stripslashes($data);   $data = htmlspecialchars($data);   return $data; } ?>

# //Form Validation HTML CODE

# <form method="post" action="<?php echo htmlspecialchars($\_SERVER["PHP\_SELF"]);?>">

# Name: <input type="text" name="name">   <span class="error">\* <?php echo $nameErr;?></span>

# <br>

# E-mail: <input type="text" name="email">   <span class="error">\* <?php echo $emailErr;?></span>

# <br>

# Website: <input type="text" name="website">   <span class="error"><?php echo $websiteErr;?></span>

# <br>

# Comment: <textarea name="comment" rows="5" cols="40"></textarea>   <br>

# Gender:   <input type="radio" name="gender" value="female">Female   <input type="radio" name="gender" value="male">Male   <input type="radio" name="gender" value="other">Other   <span class="error">\* <?php echo $genderErr;?></span>   <br>

# <input type="submit" name="submit" value="Submit">

# </form>

PHP **Date and Time**

## The PHP Date() Function

* **date(format,timestamp)**

|  |  |
| --- | --- |
| **Parameter** | **Description** |
| Format | Required. Specifies the format of the timestamp |
| Timestamp | Specifies a timestamp. Default is the current date and time |

## Get a Date

* d - Represents the day of the month (01 to 31)
* m - Represents a month (01 to 12)
* Y - Represents a year (in four digits)
* l (lowercase 'L') - Represents the day of the week

Other characters, like"/", ".", or "-" can also be inserted between the characters to add additional formatting.

|  |  |
| --- | --- |
| <?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"); ?> | Today is 2019/11/19 Today is 2019.11.19 Today is 2019-11-19 Today is Tuesday |

|  |  |
| --- | --- |
| **mktime(hour, minute, second, month, day, year)** | |
| <?php $d=mktime(11, 14, 54, 8, 12, 2014);echo "Created date is " . date("Y-m-d h:i:sa", $d); ?> | Created date is 2014-08-12 11:14:54am |

Get a Time

Here are some characters that are commonly used for times:

* H - 24-hour format of an hour (00 to 23)
* h - 12-hour format of an hour with leading zeros (01 to 12)
* i - Minutes with leading zeros (00 to 59)
* s - Seconds with leading zeros (00 to 59)
* a - Lowercase Ante meridiem and Post meridiem (am or pm)

|  |  |
| --- | --- |
| <?php echo "The time is " . date("h:i:sa"); ?> | The time is 04:27:17pm |

## Get Your Time Zone

|  |  |
| --- | --- |
| <?php date\_default\_timezone\_set("asia/dhaka"); echo "The time is " . date("h:i:sa"); ?> | The time is 11:28:36am |

Cate a Date From a String With strtotime()

**The strtotime() function parses an English textual datetime into a Unix timestamp (the number of seconds since January 1 1970 00:00:00 GMT).**

**Note: If the year is specified in a two-digit format, values between 0-69 are mapped to 2000-2069 and values between 70-100 are mapped to 1970-2000**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| <?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>"; ?> | | | 2019-11-20 12:00:00am 2019-11-23 12:00:00am 2020-02-19 04:31:03pm | | |
| More Date Examples | | | | | |
| <?php $startdate = strtotime("Saturday"); $enddate = strtotime("+6 weeks", $startdate);  while ($startdate < $enddate) {   echo date("M d", $startdate) . "<br>";   $startdate = strtotime("+1 week", $startdate);}?> | | | | Nov 23 Nov 30 Dec 07 Dec 14 Dec 21 Dec 28 | |
| outputs the number of days until 4th of July: | | | |
| $d1=strtotime("July 04"); $d2=ceil(($d1-time())/60/60/24); echo "There are " . $d2 ." days until 4th of July." | There are -138 days until 4th of July. | | |

PHP **Include Files**

include and require statements are identical, except upon failure:

* require will produce a fatal error (E\_COMPILE\_ERROR) and stop the script
* include will only produce a warning (E\_WARNING) and the script will continue
* **Use require when the file is required by the application.**
* **Use include when the file is not required and application should continue when file is not found.**
* **Include\_once & require\_once () function works once time only.**

|  |  |
| --- | --- |
| <h1>Welcome to my home page!</h1> <p>Some text.</p> <p>Some more text.</p> <?php include 'footer.php';?> | Welcome to my home page! Some text.  Some more text.  Copyright © 1999-2019 W3Schools.com |

**PHP File Open/Read/Close**

A better method to open files is with the fopen() function. This function gives you more options than the readfile() function.

|  |  |
| --- | --- |
| **Mod** | **Description** |
| R | **Open a file for read only**. File pointer starts at the beginning of the file |
| W | **Open a file for write only**. Erases the contents of the file or creates a new file if it doesn't exist. File pointer starts at the beginning of the file |
| A | **Open a file for write only**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist |
| X | **Creates a new file for write only**. Returns FALSE and an error if file already exists |
| r+ | **Open a file for read/write**. File pointer starts at the beginning of the file |
| w+ | **Open a file for read/write**. Erases the contents of the file or creates a new file if it doesn't exist. File pointer starts at the beginning of the file |
| a+ | **Open a file for read/write**. The existing data in file is preserved. File pointer starts at the end of the file. Creates a new file if the file doesn't exist |
| x+ | **Creates a new file for read/write**. Returns FALSE and an error if file already exists |

PHP Open File - fopen()

The first parameter of fopen() contains the name of the file to be opened and the second parameter specifies in which mode the file should be opened.

|  |
| --- |
| <?php $myfile = fopen("webdictionary.txt", "r") or die("Unable to open file!"); echo fread($myfile,filesize("webdictionary.txt")); fclose($myfile); ?> |
| HP = PHP Hypertext Preprocessor SQL = Structured Query Language SVG = Scalable Vector Graphics XML = EXtensible Markup Language |

PHP Read File - fread()

The fread() function reads from an open file.

The first parameter of fread() contains the name of the file to read from and the second parameter specifies the maximum number of bytes to read.

fread($myfile,filesize("webdictionary.txt"));

PHP **File Upload**

However, with ease comes danger, so always be careful when allowing file uploads!

Configure The "php.ini" File

In your "php.ini" file, search for the file\_uploads directive, and set it to On: File\_Uplods = ON

Create The HTML Form

<form action="upload.php" method="post"

enctype="multipart/form-data">

//Select Image to Upload:

<input type="file" name="fileToUpload"

<input type="submit" value="Upload Image" name="submit">  
</form>

**Some rules to follow for the HTML form above:**

* Make sure that the form uses method="post"
* The form also needs the following attribute: enctype="multipart/form-data". It specifies which content-type to use when submitting the form

Without the requirements above, the file upload will not work.

**Other things to notice:**

* The type="file" attribute of the <input> tag shows the input field as a file-select control, with a "Browse" button next to the input control

enctype='multipart/form-data is an encoding type that allows files to be sent through a POST. Quite simply, without this encoding the files cannot be sent through POST. If you want to allow a user to upload a file via a form, you must use this enctype*.*

|  |
| --- |
| PHP File UPLOAD |
| <form method="POST" action="upload.php"  enctype="multipart/form-data">  <input type="file" name="file">  <button type="submit" name="submit" >UPLOAD </button>  </form> |
| \if (isset($\_POST['submit'])) {  $file= $\_FILES["file"];  echo "<pre>";  print\_r($\_FILES);  echo "</pre>";  $fname = $\_FILES['file'] ['name'];  $ftype = $\_FILES['file'] ['type'];  $ftmp = $\_FILES ['file'] ['tmp\_name'];  $ferror = $\_FILES ['file']['error'];  $fsize = $\_FILES ['file']['size'];  $fext = pathinfo($fname, PATHINFO\_EXTENSION);  echo $fname . "<br>";  echo $ftype."<br>";  echo $ftmp."<br>";  echo $ferror."<br>";  echo $fsize."<br>";  move\_uploaded\_file($ftmp,'upload/'. $fname);  } |
| Array  ( [file] => Array  ([name] => Responsive\_Web\_Design.png  [type] => image/png  [tmp\_name] => E:\xampp\tmp\phpD5D4.tmp  [error] => 0  [size] => 118195  ))  File Details:  Responsive\_Web\_Design.png image/png E:\xampp\tmp\php95C3.tmp 0 118195 |

* In this case **isset($\_POST['submit**']) checks that the $\_**POST['submit**'] is set or not. it will be set **if** the user click on the login button.
* **print\_r in php**. It is an inbuilt function that is used in **PHP** to print or display the information stored in a variable.
* **The move\_uploaded\_file()** function moves an uploaded file to a new destination. **Note:** This function only works on files uploaded via PHP's HTTP POST upload mechanism. **Note:** If the destination file already exists, it will be overwritten.
* Avoid Overwriting exciting files you can use **date in PHP function()**

## move\_uploaded\_file($ftmp,'upload/'. date('d\_m\_Y\_h\_i\_s '). $fname);

Define extension

$fext = pathinfo($fname, PATHINFO\_EXTENSION);

//you can find your file extension with this founction……

echo ‘your file format is: ‘ .$text;//will execute your file format.

Byte\_KB\_MB

* 1=>One Byte.
* 10=>Ten Byte.
* 100=>Hundred Byte.
* 1000=>Thousand Byte / One KB.
* 10000=>Ten KB.
* 100000=>Hundred KB.
* 1000000=>Thousand KB / One MB.

Though 1024 kb for 1 MB. This for Calculation Facility.

Complete File Upload With Condition:

<?php

if (isset($\_POST['submit'])) {

//echo "<pre>";

//print\_r($\_FILES);

//echo "</pre>";

$f\_name = $\_FILES['myfile']['name'];

$f\_type = $\_FILES['myfile']['type'];

$f\_tmp = $\_FILES['myfile']['tmp\_name'];

$f\_size = $\_FILES['myfile']['size'];

$f\_error= $\_FILES['myfile']['error'];

$f\_ext = strtolower(pathinfo($f\_name,PATHINFO\_EXTENSION));

if (!file\_exists('upload/'. $f\_name)) {

if ($f\_size<1000000) {

if ($f\_ext == 'png' or $f\_ext =='jpg') {

move\_uploaded\_file($f\_tmp, 'upload/'.$f\_name);

echo "congreats your file is successfully uploaded";

}else{ echo"Unsupported File Format. Please Upload a file png or jpg Format";}

else{echo "Your File is Too Large";}

}else{

echo "Your File is already exist";}}?>

<form enctype="multipart/form-data" method="POST" action="<?php $\_SERVER['PHP\_SELF'] ?>" >

<input type="file" name="myfile">

<button type="submit" name="submit" >UPLOAD</button>

</form>

PHP COOKIE

A cookie is a small file that the server embeds on the user's computer. Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

* **Website Automatically Suggest your Previous Search items by Cookie.**
* **It will be set till the expire time of your setcookie function.**
* **Your Can Also Delete cookie time() - 3600);**

A cookie is created with the setcookie() function.

**setcookie(name, value, expire, path, domain, secure, httponly);**

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

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

**//Create Cookie....**

**//86400 seconds = 1 day....**

**//if (isset($\_POST['submit']))**

**{setcookie('Cloth', $\_POST['shirt'], time()+ 86400,"/");}**

**//Check Cooktie...**

**if (isset($\_POST['submit'])) {**

**if (count($\_COOKIE)>0) {**

**echo "You Already Have ". count($\_COOKIE). "Cookies";**

**} else{setcookie('cloth', $\_POST['shirt'], time()+ 86400);}**

**}**

**//Delete Cookie.....**

**//setcookie('Cloth', $\_POST['shirt'], time() - 14400);**

PHP Session

**A session is a way to store information (in variables) to be used across multiple pages.**

Unlike a cookie, the information is not stored on the user’s computer.

What is a PHP Session?

When you work with an application, you open it, do some changes, and then you close it. This is much like a Session. The computer knows who you are. It knows when you start the application and when you end.

**Tip: If you need a permanent storage, you may want to store the data in a**[**database**](https://www.w3schools.com/php/php_mysql_intro.asp)**.**

* **A session is started with the session\_start() function.**

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

* **session\_start() function must be before HTML tags.**
* session\_unset() and session\_destroy():

<?php  
// remove all session variables  
session\_unset();  
  
// destroy the session  
session\_destroy();  
?

PAGE TRANSFAR IN PHP

header('location:profile.php');

**PHP FILTERS**

* **Validating data = Determine if the data is in proper form.**
* **Sanitizing data = Remove any illegal character from the data.**

**You should always validate external data!**Invalid submitted data can lead to security problems and break your webpage!  
By using PHP filters you can be sure your application gets the correct input!

## PHP filter\_var() Function

The filter\_var() function both validate and sanitize data.

The filter\_var() function filters a single variable with a specified filter. It takes two pieces of data:

* The variable you want to check
* The type of check to use

|  |  |
| --- | --- |
| <?php $str = "<h1>Hello World!</h1>"; $newstr = filter\_var($str,FILTER\_SANITIZE\_STRING); echo $newstr; ?> | //Normal Script  Hello World! |

Sanitize and Validate a URL

|  |  |
| --- | --- |
| <?php $url = "https://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"); } ?> | https://www.w3schools.com is a valid URL |

Validate IP

|  |  |
| --- | --- |
| <?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"); } ?> | 127.0.0.1  is a valid IP address |

Filter Advance

### Filter Range

|  |  |
| --- | --- |
| INPUT  <?php  $num = 200;  $max = 100;  $min= 1;  if (filter\_var($num, FILTER\_VALIDATE\_INT,  array('options'=>array('min\_range'=>$min, 'max\_range'=>$max))))  { echo "The INT Value Is valid";  }else{echo "The int value is unvalid";}  ?> | OUTPUT  Valid Value |

### Filter Query

|  |  |
| --- | --- |
| INPUT  <?php  $url = 'http://marketernazmul.com/?php';  if (filter\_var($url, FILTER\_VALIDATE\_URL, FILTER\_FLAG\_QUERY\_REQUIRED))  {echo 'This URL is query Included';  }else{ echo "This URL Is Query Included";}  ?> | OUTPUT  This Url is Query Included |

Explode & Implode Function

* **Explode Function Converts String to array.**
* **Implode Function Converts array to string.**

## Explode Syntax: “explode(separator, string);”

|  |  |
| --- | --- |
| <?php  $x = 'I Love PHP Programming';  $y = explode(' ', $x);  foreach ($y as $value) {  echo $value;  echo '<br>';  }?> | I  Love  PHP  Programming |

## Implode Syntax: Implode (separator, array);

|  |  |
| --- | --- |
| <?php  $name= array('nazmul','billal','mohammod');  echo implode(' ', $name);  ?> | Nazmul billal mohammod |

Sentence Case:

* **Strtoupper()-> String To Uppercase.**
* **Strtolower()-> String To Lower.**
* **Ucfirst()-> User First character of sentence.**
* **Ucwords()->User each**