PHP Introduction

[❮ Previous](https://www.w3schools.com/php/default.asp)[Next ❯](https://www.w3schools.com/php/php_install.asp)

PHP code is executed on the server.

What You Should Already Know

Before you continue you should have a basic understanding of the following:

* [HTML](https://www.w3schools.com/html/default.asp)
* [CSS](https://www.w3schools.com/css/default.asp)
* [JavaScript](https://www.w3schools.com/js/default.asp)

If you want to study these subjects first, find the tutorials on our [Home page](https://www.w3schools.com/default.asp).

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 large social networks!  
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

With PHP you are not limited to output HTML. You can output images or PDF files. You can also output any text, such as XHTML and XML.

ADVERTISEMENT

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: <=>)

What Do I Need?

To start using PHP, you can:

* Find a web host with PHP and MySQL support
* Install a web server on your own PC, and then install PHP and MySQL

Use a Web Host With PHP Support

If your server has activated support for PHP you do not need to do anything.

Just create some .php files, place them in your web directory, and the server will automatically parse them for you.

You do not need to compile anything or install any extra tools.

Because PHP is free, most web hosts offer PHP support.

Set Up PHP on Your Own PC

However, if your server does not support PHP, you must:

* install a web server
* install PHP
* install a database, such as MySQL

The official PHP website (PHP.net) has installation instructions for PHP: <http://php.net/manual/en/install.php>

PHP Online Compiler / Editor

With w3schools' online PHP compiler, you can edit PHP code, and view the result in your browser.

[Run »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_compiler)

<?php  
$txt = "PHP";  
echo "I love $txt!";  
?>

I love PHP!

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_compiler)

Click on the "Try it Yourself" button to see how it works.

# PHP Syntax

[❮ Previous](https://www.w3schools.com/php/php_install.asp)[Next ❯](https://www.w3schools.com/php/php_comments.asp)

A PHP script is executed on the server, and the plain HTML result is sent back to the browser.

## 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".

A PHP file normally contains HTML tags, and some PHP scripting code.

Below, we have an example of a simple PHP file, with a PHP script that uses a built-in PHP function "echo" to output the text "Hello World!" on a web page:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<!DOCTYPE html>  
<html>  
<body>  
  
<h1>My first PHP page</h1>  
  
<?php  
echo "Hello World!";  
?>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_syntax)

**Note:** PHP statements end with a semicolon (;).

ADVERTISEMENT

## PHP Case Sensitivity

In PHP, keywords (e.g. if, else, while, echo, etc.), classes, functions, and user-defined functions are not case-sensitive.

In the example below, all three echo statements below are equal and legal:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_syntax_case1)

**Note:** However; all variable names are case-sensitive!

Look at the example below; only the first statement will display the value of the $color variable! This is because $color, $COLOR, and $coLOR are treated as three different variables:

### **Example**

<!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>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_syntax_case2)

# PHP Comments

[❮ Previous](https://www.w3schools.com/php/php_syntax.asp)[Next ❯](https://www.w3schools.com/php/php_variables.asp)

## Comments in PHP

A comment in PHP code is a line that is not executed as a part of the program. Its only purpose is to be read by someone who is looking at the code.

Comments can be used to:

* Let others understand your code
* Remind yourself of what you did - Most programmers have experienced coming back to their own work a year or two later and having to re-figure out what they did. Comments can remind you of what you were thinking when you wrote the code

PHP supports several ways of commenting:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Syntax for single-line comments:

<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
// This is a single-line comment  
  
# This is also a single-line comment  
?>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_comments)

### **Example**

Syntax for multiple-line comments:

<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
/\*  
This is a multiple-lines comment block  
that spans over multiple  
lines  
\*/  
?>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_comments2)

### **Example**

Using comments to leave out parts of the code:

<!DOCTYPE html>  
<html>  
<body>  
  
<?php  
// You can also use comments to leave out parts of a code line  
$x = 5 /\* + 15 \*/ + 5;  
echo $x;  
?>  
  
</body>  
</html>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_comments3)

# PHP Variables

[❮ Previous](https://www.w3schools.com/php/php_comments.asp)[Next ❯](https://www.w3schools.com/php/php_variables_scope.asp)

Variables are "containers" for storing information.

## Creating (Declaring) PHP Variables

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

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
$txt = "Hello world!";  
$x = 5;  
$y = 10.5;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var)

After the execution of the statements above, the variable $txt will hold the value Hello world!, the variable $x will hold the value 5, and the variable $y will hold the value 10.5.

**Note:** When you assign a text value to a variable, put quotes around the value.

**Note:** Unlike other programming languages, PHP has no command for declaring a variable. It is created the moment you first assign a value to it.

Think of variables as containers for storing data.

## 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)

Remember that PHP variable names are case-sensitive!

ADVERTISEMENT

## Output Variables

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

The following example will show how to output text and a variable:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var2)

The following example will produce the same output as the example above:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var3)

The following example will output the sum of two variables:

### **Example**

<?php  
$x = 5;  
$y = 4;  
echo $x + $y;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var4)

**Note:** You will learn more about the echo statement and how to output data to the screen in the next chapter.

## PHP is a Loosely Typed Language

In the example above, notice that we did not have to tell PHP which data type the variable is.

PHP automatically associates a data type to the variable, depending on its value. Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error.

In PHP 7, type declarations were added. This gives an option to specify the data type expected when declaring a function, and by enabling the strict requirement, it will throw a "Fatal Error" on a type mismatch.

You will learn more about strict and non-strict requirements, and data type declarations in the [PHP Functions](https://www.w3schools.com/php/php_functions.asp) chapter.

# PHP Variables Scope

[❮ Previous](https://www.w3schools.com/php/php_variables.asp)[Next ❯](https://www.w3schools.com/php/php_echo_print.asp)

## PHP Variables Scope

In PHP, variables can be declared anywhere in the script.

The scope of a variable is the part of the script where the variable can be referenced/used.

PHP has three different variable scopes:

* local
* global
* static

## Global and Local Scope

A variable declared **outside** a function has a GLOBAL SCOPE and can only be accessed outside a function:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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>";  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var_global)

A variable declared **within** a function has a LOCAL SCOPE and can only be accessed within that function:

### **Example**

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>";  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var_local)

You can have local variables with the same name in different functions, because local variables are only recognized by the function in which they are declared.

ADVERTISEMENT

## PHP The global Keyword

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

To do this, use the global keyword before the variables (inside the function):

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var_global_keyword)

PHP also stores all global variables in an array called $GLOBALS[*index*]. The *index* holds the name of the variable. This array is also accessible from within functions and can be used to update global variables directly.

The example above can be rewritten like this:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var_globals)

## PHP The static Keyword

Normally, when a function is completed/executed, all of its variables are deleted. However, sometimes we want a local variable NOT to be deleted. We need it for a further job.

To do this, use the static keyword when you first declare the variable:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_var_static)

Then, each time the function is called, that variable will still have the information it contained from the last time the function was called.

**Note:** The variable is still local to the function.

# PHP echo and print Statements

[❮ Previous](https://www.w3schools.com/php/php_variables_scope.asp)[Next ❯](https://www.w3schools.com/php/php_datatypes.asp)

With PHP, there are two basic ways to get output: echo and print.

In this tutorial we use echo or print in almost every example. So, this chapter contains a little more info about those two output statements.

## PHP echo and print Statements

echo and print are more or less the same. They are both used to output data to the screen.

The differences are small: 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.

## The PHP echo Statement

The echo statement can be used with or without parentheses: echo or echo().

**Display Text**

The following example shows how to output text with the echo command (notice that the text can contain HTML markup):

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?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.";  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_echo1)

**Display Variables**

The following example shows how to output text and variables with the echo statement:

### **Example**

<?php  
$txt1 = "Learn PHP";  
$txt2 = "W3Schools.com";  
$x = 5;  
$y = 4;  
  
echo "<h2>" . $txt1 . "</h2>";  
echo "Study PHP at " . $txt2 . "<br>";  
echo $x + $y;  
?>

# PHP Data Types

[❮ Previous](https://www.w3schools.com/php/php_echo_print.asp)[Next ❯](https://www.w3schools.com/php/php_string.asp)

## PHP Data Types

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

PHP supports the following data types:

* 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!".

A string can be any text inside quotes. You can use single or double quotes:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_string)

## PHP Integer

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

Rules for integers:

* An integer must have at least one digit
* An integer must not have a decimal point
* An integer can be either positive or negative
* Integers can be specified in: decimal (base 10), hexadecimal (base 16), octal (base 8), or binary (base 2) notation

In the following example $x is an integer. The PHP var\_dump() function returns the data type and value:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_integer)

ADVERTISEMENT

## PHP Float

A float (floating point number) is a number with a decimal point or a number in exponential form.

In the following example $x is a float. The PHP var\_dump() function returns the data type and value:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_float)

## PHP Boolean

A Boolean represents two possible states: TRUE or FALSE.

$x = true;  
$y = false;

Booleans are often used in conditional testing. You will learn more about conditional testing in a later chapter of this tutorial.

## PHP Array

An array stores multiple values in one single variable.

In the following example $cars is an array. The PHP var\_dump() function returns the data type and value:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_array)

You will learn a lot more about arrays in later chapters of this tutorial.

## PHP Object

Classes and objects are the two main aspects of object-oriented programming.

A class is a template for objects, and an object is an instance of a class.

When the individual objects are created, they inherit all the properties and behaviors from the class, but each object will have different values for the properties.

Let's assume we have a class named Car. A Car can have properties like model, color, etc. We can define variables like $model, $color, and so on, to hold the values of these properties.

When the individual objects (Volvo, BMW, Toyota, etc.) are created, they inherit all the properties and behaviors from the class, but each object will have different values for the properties.

If you create a \_\_construct() function, PHP will automatically call this function when you create an object from a class.

### **Example**

<?php  
class Car {  
  public $color;  
  public $model;  
  public function \_\_construct($color, $model) {  
    $this->color = $color;  
    $this->model = $model;  
  }  
  public function message() {  
    return "My car is a " . $this->color . " " . $this->model . "!";  
  }  
}  
  
$myCar = new Car("black", "Volvo");  
echo $myCar -> message();  
echo "<br>";  
$myCar = new Car("red", "Toyota");  
echo $myCar -> message();  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_object)

## PHP NULL Value

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.

Variables can also be emptied by setting the value to NULL:

### **Example**

<?php  
$x = "Hello world!";  
$x = null;  
var\_dump($x);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_datatypes_null)

## PHP Resource

The special resource type is not an actual data type. It is the storing of a reference to functions and resources external to PHP.

A common example of using the resource data type is a database call.

We will not talk about the resource type here, since it is an advanced topic.

# PHP Strings

[❮ Previous](https://www.w3schools.com/php/php_datatypes.asp)[Next ❯](https://www.w3schools.com/php/php_numbers.asp)

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

## PHP String Functions

In this chapter we will look at some commonly used functions to manipulate strings.

## strlen() - Return the Length of a String

The PHP strlen() function returns the length of a string.

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Return the length of the string "Hello world!":

<?php  
echo strlen("Hello world!"); // outputs 12  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_string_length)

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

The PHP str\_word\_count() function counts the number of words in a string.

### **Example**

Count the number of word in the string "Hello world!":

<?php  
echo str\_word\_count("Hello world!"); // outputs 2  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_string_word_count)

ADVERTISEMENT

## strrev() - Reverse a String

The PHP strrev() function reverses a string.

### **Example**

Reverse the string "Hello world!":

<?php  
echo strrev("Hello world!"); // outputs !dlrow olleH  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_string_reverse)

## strpos() - Search For a Text Within a String

The PHP strpos() function searches for a specific text within a string. If a match is found, the function returns the character position of the first match. If no match is found, it will return FALSE.

### **Example**

Search for the text "world" in the string "Hello world!":

<?php  
echo strpos("Hello world!", "world"); // outputs 6  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_string_pos)

**Tip:** The first character position in a string is 0 (not 1).

## str\_replace() - Replace Text Within a String

The PHP str\_replace() function replaces some characters with some other characters in a string.

### **Example**

Replace the text "world" with "Dolly":

<?php  
echo str\_replace("world", "Dolly", "Hello world!"); // outputs Hello Dolly!  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_string_replace)

## Complete PHP String Reference

For a complete reference of all string functions, go to our complete [PHP String Reference](https://www.w3schools.com/php/php_ref_string.asp).

The PHP string reference contains description and example of use, for each function!

# PHP Numbers

[❮ Previous](https://www.w3schools.com/php/php_string.asp)[Next ❯](https://www.w3schools.com/php/php_math.asp)

In this chapter we will look in depth into Integers, Floats, and Number Strings.

## PHP Numbers

One thing to notice about PHP is that it provides automatic data type conversion.

So, if you assign an integer value to a variable, the type of that variable will automatically be an integer. Then, if you assign a string to the same variable, the type will change to a string.

This automatic conversion can sometimes break your code.

## PHP Integers

2, 256, -256, 10358, -179567 are all integers.

An integer is a number without any decimal part.

An integer data type is a non-decimal number between -2147483648 and 2147483647 in 32 bit systems, and between -9223372036854775808 and 9223372036854775807 in 64 bit systems. A value greater (or lower) than this, will be stored as float, because it exceeds the limit of an integer.

**Note:** Another important thing to know is that even if 4 \* 2.5 is 10, the result is stored as float, because one of the operands is a float (2.5).

Here are some rules for integers:

* An integer must have at least one digit
* An integer must NOT have a decimal point
* An integer can be either positive or negative
* Integers can be specified in three formats: decimal (10-based), hexadecimal (16-based - prefixed with 0x) or octal (8-based - prefixed with 0)

PHP has the following predefined constants for integers:

* PHP\_INT\_MAX - The largest integer supported
* PHP\_INT\_MIN - The smallest integer supported
* PHP\_INT\_SIZE -  The size of an integer in bytes

PHP has the following functions to check if the type of a variable is integer:

* is\_int()
* is\_integer() - alias of is\_int()
* is\_long() - alias of is\_int()

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Check if the type of a variable is integer:

<?php  
$x = 5985;  
var\_dump(is\_int($x));  
  
$x = 59.85;  
var\_dump(is\_int($x));  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_integer)

ADVERTISEMENT

## PHP Floats

A float is a number with a decimal point or a number in exponential form.

2.0, 256.4, 10.358, 7.64E+5, 5.56E-5 are all floats.

The float data type can commonly store a value up to 1.7976931348623E+308 (platform dependent), and have a maximum precision of 14 digits.

PHP has the following predefined constants for floats (from PHP 7.2):

* PHP\_FLOAT\_MAX - The largest representable floating point number
* PHP\_FLOAT\_MIN - The smallest representable positive floating point number
* PHP\_FLOAT\_DIG - The number of decimal digits that can be rounded into a float and back without precision loss
* PHP\_FLOAT\_EPSILON - The smallest representable positive number x, so that x + 1.0 != 1.0

PHP has the following functions to check if the type of a variable is float:

* is\_float()
* is\_double() - alias of is\_float()

### **Example**

Check if the type of a variable is float:

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_float)

## PHP Infinity

A numeric value that is larger than PHP\_FLOAT\_MAX is considered infinite.

PHP has the following functions to check if a numeric value is finite or infinite:

* [is\_finite()](https://www.w3schools.com/php/func_math_is_finite.asp)
* [is\_infinite()](https://www.w3schools.com/php/func_math_is_infinite.asp)

However, the PHP var\_dump() function returns the data type and value:

### **Example**

Check if a numeric value is finite or infinite:

<?php  
$x = 1.9e411;  
var\_dump($x);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_infinite)

## PHP NaN

NaN stands for Not a Number.

NaN is used for impossible mathematical operations.

PHP has the following functions to check if a value is not a number:

* [is\_nan()](https://www.w3schools.com/php/func_math_is_nan.asp)

However, the PHP var\_dump() function returns the data type and value:

### **Example**

Invalid calculation will return a NaN value:

<?php  
$x = acos(8);  
var\_dump($x);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_nan)

## PHP Numerical Strings

The PHP is\_numeric() function can be used to find whether a variable is numeric. The function returns true if the variable is a number or a numeric string, false otherwise.

### **Example**

Check if the variable is numeric:

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_numeric)

**Note:** From PHP 7.0: The is\_numeric() function will return FALSE for numeric strings in hexadecimal form (e.g. 0xf4c3b00c), as they are no longer considered as numeric strings.

## PHP Casting Strings and Floats to Integers

Sometimes you need to cast a numerical value into another data type.

The (int), (integer), or intval() function are often used to convert a value to an integer.

### **Example**

Cast float and string to integer:

<?php  
// 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;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_numbers_cast)

# PHP Math

[❮ Previous](https://www.w3schools.com/php/php_numbers.asp)[Next ❯](https://www.w3schools.com/php/php_constants.asp)

PHP has a set of math functions that allows you to perform mathematical tasks on numbers.

## PHP pi() Function

The pi() function returns the value of PI:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
echo(pi()); // returns 3.1415926535898  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_pi)

## PHP min() and max() Functions

The min() and max() functions can be used to find the lowest or highest value in a list of arguments:

### **Example**

<?php  
echo(min(0, 150, 30, 20, -8, -200));  // returns -200  
echo(max(0, 150, 30, 20, -8, -200));  // returns 150  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_min_max)

ADVERTISEMENT

## PHP abs() Function

The abs() function returns the absolute (positive) value of a number:

### **Example**

<?php  
echo(abs(-6.7));  // returns 6.7  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_abs)

## PHP sqrt() Function

The sqrt() function returns the square root of a number:

### **Example**

<?php  
echo(sqrt(64));  // returns 8  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_sqrt)

## PHP round() Function

The round() function rounds a floating-point number to its nearest integer:

### **Example**

<?php  
echo(round(0.60));  // returns 1  
echo(round(0.49));  // returns 0  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_round)

## Random Numbers

The rand() function generates a random number:

### **Example**

<?php  
echo(rand());  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_rand)

To get more control over the random number, you can add the optional min and max parameters to specify the lowest integer and the highest integer to be returned.

For example, if you want a random integer between 10 and 100 (inclusive), use rand(10, 100):

### **Example**

<?php  
echo(rand(10, 100));  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_math_rand2)

## Complete PHP Math Reference

For a complete reference of all math functions, go to our complete [PHP Math Reference](https://www.w3schools.com/php/php_ref_math.asp).

The PHP math reference contains description and example of use, for each function.

# PHP Constants

[❮ Previous](https://www.w3schools.com/php/php_math.asp)[Next ❯](https://www.w3schools.com/php/php_operators.asp)

Constants are like variables, except that once they are defined they cannot be changed or undefined.

## 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).

**Note:** Unlike variables, constants are automatically global across the entire script.

## Create a PHP Constant

To create a constant, use the define() function.

### **Syntax**

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

Parameters:

* *name*: Specifies the name of the constant
* *value*: Specifies the value of the constant
* *case-insensitive*: Specifies whether the constant name should be case-insensitive. Default is false. **Note:** Defining case-insensitive constants was deprecated in PHP 7.3. PHP 8.0 accepts only false, the value true will produce a warning.

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Create a constant with a **case-sensitive** name:

<?php  
define("GREETING", "Welcome to W3Schools.com!");  
echo GREETING;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_constant1)

### **Example**

Create a constant with a **case-insensitive** name:

<?php  
define("GREETING", "Welcome to W3Schools.com!", true);  
echo greeting;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_constant2)

ADVERTISEMENT

## PHP Constant Arrays

In PHP7, you can create an Array constant using the define() function.

### **Example**

Create an Array constant:

<?php  
define("cars", [  
  "Alfa Romeo",  
  "BMW",  
  "Toyota"  
]);  
echo cars[0];  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_constant_array)

## Constants are Global

Constants are automatically global and can be used across the entire script.

### **Example**

This example uses a constant inside a function, even if it is defined outside the function:

<?php  
define("GREETING", "Welcome to W3Schools.com!");  
  
function myTest() {  
  echo GREETING;  
}  
   
myTest();  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_constant3)

PHP Operators

[❮ Previous](https://www.w3schools.com/php/php_constants.asp)[Next ❯](https://www.w3schools.com/php/php_if_else.asp)

PHP Operators

Operators are used to perform operations on variables and values.

PHP divides the operators in the following groups:

* Arithmetic operators
* Assignment operators
* Comparison operators
* Increment/Decrement operators
* Logical operators
* String operators
* Array operators
* Conditional assignment operators

PHP Arithmetic Operators

The PHP arithmetic operators are used with numeric values to perform common arithmetical operations, such as addition, subtraction, multiplication etc.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| + | Addition | $x + $y | Sum of $x and $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_addition) |
| - | Subtraction | $x - $y | Difference of $x and $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_subtraction) |
| \* | Multiplication | $x \* $y | Product of $x and $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_multiplication) |
| / | Division | $x / $y | Quotient of $x and $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_division) |
| % | Modulus | $x % $y | Remainder of $x divided by $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_modulus) |
| \*\* | Exponentiation | $x \*\* $y | Result of raising $x to the $y'th power | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_exponentiation) |

PHP Assignment Operators

The PHP assignment operators are used with numeric values to write a value to a variable.

The basic assignment operator in PHP is "=". It means that the left operand gets set to the value of the assignment expression on the right.

|  |  |  |  |
| --- | --- | --- | --- |
| **Assignment** | **Same as...** | **Description** | **Show it** |
| x = y | x = y | The left operand gets set to the value of the expression on the right | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_set) |
| x += y | x = x + y | Addition | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_addition2) |
| x -= y | x = x - y | Subtraction | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_subtraction2) |
| x \*= y | x = x \* y | Multiplication | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_multiplication2) |
| x /= y | x = x / y | Division | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_division2) |
| x %= y | x = x % y | Modulus | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_modulus2) |

ADVERTISEMENT

PHP Comparison Operators

The PHP comparison operators are used to compare two values (number or string):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| == | Equal | $x == $y | Returns true if $x is equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_equal) |
| === | Identical | $x === $y | Returns true if $x is equal to $y, and they are of the same type | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_identical) |
| != | Not equal | $x != $y | Returns true if $x is not equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_not_equal) |
| <> | Not equal | $x <> $y | Returns true if $x is not equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_not_equal2) |
| !== | Not identical | $x !== $y | Returns true if $x is not equal to $y, or they are not of the same type | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_not_identical) |
| > | Greater than | $x > $y | Returns true if $x is greater than $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_greater_than) |
| < | Less than | $x < $y | Returns true if $x is less than $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_less_than) |
| >= | Greater than or equal to | $x >= $y | Returns true if $x is greater than or equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_greater_than2) |
| <= | Less than or equal to | $x <= $y | Returns true if $x is less than or equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_less_than2) |
| <=> | Spaceship | $x <=> $y | Returns an integer less than, equal to, or greater than zero, depending on if $x is less than, equal to, or greater than $y. Introduced in PHP 7. | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_spaceship) |

PHP Increment / Decrement Operators

The PHP increment operators are used to increment a variable's value.

The PHP decrement operators are used to decrement a variable's value.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Description** | **Show it** |
| ++$x | Pre-increment | Increments $x by one, then returns $x | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_pre_incr) |
| $x++ | Post-increment | Returns $x, then increments $x by one | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_post_incr) |
| --$x | Pre-decrement | Decrements $x by one, then returns $x | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_pre_decr) |
| $x-- | Post-decrement | Returns $x, then decrements $x by one | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_post_decr) |

PHP Logical Operators

The PHP logical operators are used to combine conditional statements.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| and | And | $x and $y | True if both $x and $y are true | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_and) |
| or | Or | $x or $y | True if either $x or $y is true | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_or) |
| xor | Xor | $x xor $y | True if either $x or $y is true, but not both | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_xor) |
| && | And | $x && $y | True if both $x and $y are true | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_and2) |
| || | Or | $x || $y | True if either $x or $y is true | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_or2) |
| ! | Not | !$x | True if $x is not true | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_not) |

PHP String Operators

PHP has two operators that are specially designed for strings.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| . | Concatenation | $txt1 . $txt2 | Concatenation of $txt1 and $txt2 | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_string1) |
| .= | Concatenation assignment | $txt1 .= $txt2 | Appends $txt2 to $txt1 | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_string2) |

PHP Array Operators

The PHP array operators are used to compare arrays.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| + | Union | $x + $y | Union of $x and $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_union) |
| == | Equality | $x == $y | Returns true if $x and $y have the same key/value pairs | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_equality) |
| === | Identity | $x === $y | Returns true if $x and $y have the same key/value pairs in the same order and of the same types | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_identity) |
| != | Inequality | $x != $y | Returns true if $x is not equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_inequality) |
| <> | Inequality | $x <> $y | Returns true if $x is not equal to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_inequality2) |
| !== | Non-identity | $x !== $y | Returns true if $x is not identical to $y | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_arr_non_identity) |

PHP Conditional Assignment Operators

The PHP conditional assignment operators are used to set a value depending on conditions:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Name** | **Example** | **Result** | **Show it** |
| ?: | Ternary | $x = *expr1* ? *expr2* : *expr3* | Returns the value of $x. The value of $x is *expr2* if *expr1* = TRUE. The value of $x is *expr3* if *expr1* = FALSE | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_ternary) |
| ?? | Null coalescing | $x = *expr1* ?? *expr2* | Returns the value of $x. The value of $x is *expr1* if *expr1* exists, and is not NULL. If *expr1* does not exist, or is NULL, the value of $x is *expr2*. Introduced in PHP 7 | [Try it »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_oper_null_coalescing) |

# PHP if...else...elseif Statements

[❮ Previous](https://www.w3schools.com/php/php_operators.asp)[Next ❯](https://www.w3schools.com/php/php_switch.asp)

Conditional statements are used to perform different actions based on different conditions.

## PHP Conditional Statements

Very often when you write code, you want to perform different actions for different conditions. You can use conditional statements in your code to do this.

In PHP we have the following conditional statements:

* 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

## PHP - The if Statement

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

### **Syntax**

if (*condition*) { *code to be executed if condition is true*;  
}

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Output "Have a good day!" if the current time (HOUR) is less than 20:

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_if)

ADVERTISEMENT

## PHP - The if...else Statement

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

### **Syntax**

if (*condition*) {  
  *code to be executed if condition is true;*  
} else {  
  *code to be executed if condition is false;*}

### **Example**

Output "Have a good day!" if the current time is less than 20, and "Have a good night!" otherwise:

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_if_else)

## PHP - The if...elseif...else Statement

The if...elseif...else statement executes different codes for more than two conditions.

### **Syntax**

if (*condition*) {  
  *code to be executed if this condition is true;*} elseif (*condition*) {  
 *code to be executed if first condition is false and this condition is true;*} else {  
  *code to be executed if all conditions are false;*}

### **Example**

Output "Have a good morning!" if the current time is less than 10, and "Have a good day!" if the current time is less than 20. Otherwise it will output "Have a good night!":

<?php  
$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!";  
}  
?>

# PHP switch Statement

[❮ Previous](https://www.w3schools.com/php/php_if_else.asp)[Next ❯](https://www.w3schools.com/php/php_looping.asp)

The switch statement is used to perform different actions based on different conditions.

## The PHP switch Statement

Use the switch statement to **select one of many blocks of code to be executed**.

### **Syntax**

switch (*n*) {  
  case *label1:*  
  *code to be executed if n=label1;*  
    break;  
  case *label2:*  
  *code to be executed if n=label2;*  
    break;  
  case *label3:*  
  *code to be executed if n=label3;*  
    break;  
    ...  
  default:  
  *code to be executed if n is different from all labels;*  
}

This is how it works: First we have a single expression *n* (most often a variable), that is evaluated once. The value of the expression is then compared with the values for each case in the structure. If there is a match, the block of code associated with that case is executed. Use break to prevent the code from running into the next case automatically. The default statement is used if no match is found.

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?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!";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_switch)

ADVERTISEMENT

PHP Loops

[❮ Previous](https://www.w3schools.com/php/php_switch.asp)[Next ❯](https://www.w3schools.com/php/php_looping_while.asp)

In the following chapters you will learn how to repeat code by using loops in PHP.

PHP Loops

Often when you write code, you want the same block of code to run over and over again a certain number of times. So, instead of adding several almost equal code-lines in a script, we can use loops.

Loops are used to execute the same block of code again and again, as long as a certain condition is true.

In PHP, we have the following loop types:

* 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 following chapters will explain and give examples of each loop type.

# PHP while Loop

[❮ Previous](https://www.w3schools.com/php/php_looping.asp)[Next ❯](https://www.w3schools.com/php/php_looping_do_while.asp)

The while loop - Loops through a block of code as long as the specified condition is true.

## The PHP while Loop

The while loop executes a block of code as long as the specified condition is true.

### **Syntax**

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

### **Examples**

The example below displays the numbers from 1 to 5:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_while)

### **Example Explained**

* $x = 1; - Initialize the loop counter ($x), and set the start value to 1
* $x <= 5 - Continue the loop as long as $x is less than or equal to 5
* $x++; - Increase the loop counter value by 1 for each iteration

This example counts to 100 by tens:

### **Example**

<?php  
$x = 0;  
  
while($x <= 100) {  
  echo "The number is: $x <br>";  
  $x+=10;  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_while2)

### **Example Explained**

* $x = 0; - Initialize the loop counter ($x), and set the start value to 0
* $x <= 100 - Continue the loop as long as $x is less than or equal to 100
* $x+=10; - Increase the loop counter value by 10 for each iteration

# PHP do while Loop

[❮ Previous](https://www.w3schools.com/php/php_looping_while.asp)[Next ❯](https://www.w3schools.com/php/php_looping_for.asp)

The do...while loop - Loops through a block of code once, and then repeats the loop as long as the specified condition is true.

## The PHP do...while Loop

The do...while loop will always execute the block of code once, it will then check the condition, and repeat the loop while the specified condition is true.

### **Syntax**

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

### **Examples**

The example below first sets a variable $x to 1 ($x = 1). Then, the do while loop will write some output, and then increment the variable $x with 1. Then the condition is checked (is $x less than, or equal to 5?), and the loop will continue to run as long as $x is less than, or equal to 5:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_do_while)

**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.

This example sets the $x variable to 6, then it runs the loop, **and then the condition is checked**:

### **Example**

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

# PHP for Loop

[❮ Previous](https://www.w3schools.com/php/php_looping_do_while.asp)[Next ❯](https://www.w3schools.com/php/php_looping_foreach.asp)

The for loop - Loops through a block of code a specified number of times.

## The PHP for Loop

The for loop is used when you know in advance how many times the script should run.

### **Syntax**

for (*init counter; test counter; increment counter*) {  
  *code to be executed for each iteration;*  
}

Parameters:

* *init counter*: Initialize the loop counter value
* *test counter*: Evaluated for each loop iteration. If it evaluates to TRUE, the loop continues. If it evaluates to FALSE, the loop ends.
* *increment counter*: Increases the loop counter value

### **Examples**

The example below displays the numbers from 0 to 10:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_for)

### **Example Explained**

* $x = 0; - Initialize the loop counter ($x), and set the start value to 0
* $x <= 10; - Continue the loop as long as $x is less than or equal to 10
* $x++ - Increase the loop counter value by 1 for each iteration

This example counts to 100 by tens:

### **Example**

<?php  
for ($x = 0; $x <= 100; $x+=10) {  
  echo "The number is: $x <br>";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_for2)

### **Example Explained**

* $x = 0; - Initialize the loop counter ($x), and set the start value to 0
* $x <= 100; - Continue the loop as long as $x is less than or equal to 100
* $x+=10 - Increase the loop counter value by 10 for each iteration

ADVERTISEMENT

# PHP foreach Loop

[❮ Previous](https://www.w3schools.com/php/php_looping_for.asp)[Next ❯](https://www.w3schools.com/php/php_looping_break.asp)

The foreach loop - Loops through a block of code for each element in an array.

## The PHP foreach Loop

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

### **Syntax**

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

For every loop iteration, the value of the current array element is assigned to $value and the array pointer is moved by one, until it reaches the last array element.

### **Examples**

The following example will output the values of the given array ($colors):

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_foreach)

The following example will output both the keys and the values of the given array ($age):

### **Example**

<?php  
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");  
  
foreach($age as $x => $val) {  
  echo "$x = $val<br>";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_loop_foreach2)

You will learn more about arrays in the [PHP Arrays](https://www.w3schools.com/php/php_arrays.asp) chapter.

# PHP Break and Continue

[❮ Previous](https://www.w3schools.com/php/php_looping_foreach.asp)[Next ❯](https://www.w3schools.com/php/php_functions.asp)

## PHP Break

You have already seen the break statement used in an earlier chapter of this tutorial. It was used to "jump out" of a switch statement.

The break statement can also be used to jump out of a loop.

This example jumps out of the loop when **x** is equal to **4**:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
for ($x = 0; $x < 10; $x++) {  
  if ($x == 4) {  
    break;  
  }  
  echo "The number is: $x <br>";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_break)

## PHP Continue

The continue statement breaks one iteration (in the loop), if a specified condition occurs, and continues with the next iteration in the loop.

This example skips the value of **4**:

### **Example**

<?php  
for ($x = 0; $x < 10; $x++) {  
  if ($x == 4) {  
    continue;  
  }  
  echo "The number is: $x <br>";  
}  
?>

## Break and Continue in While Loop

You can also use break and continue in while loops:

### **Break Example**

<?php  
$x = 0;  
  
while($x < 10) {  
  if ($x == 4) {  
    break;  
  }  
  echo "The number is: $x <br>";  
  $x++;  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_break_while)

### **Continue Example**

<?php  
$x = 0;  
  
while($x < 10) {  
  if ($x == 4) {  
    $x++;  
    continue;  
  }  
  echo "The number is: $x <br>";  
  $x++;  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_continue_while)

[❮ Previous](https://www.w3schools.com/php/php_looping_foreach.asp)[Next ❯](https://www.w3schools.com/php/php_functions.asp)

# PHP Functions

[❮ Previous](https://www.w3schools.com/php/php_looping_break.asp)[Next ❯](https://www.w3schools.com/php/php_arrays.asp)

The real power of PHP comes from its functions.

PHP has more than 1000 built-in functions, and in addition you can create your own custom functions.

## PHP Built-in Functions

PHP has over 1000 built-in functions that can be called directly, from within a script, to perform a specific task.

Please check out our PHP reference for a complete overview of the [PHP built-in functions](https://www.w3schools.com/php/php_ref_overview.asp).

## PHP User Defined Functions

Besides the built-in PHP functions, it is possible to create your own functions.

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

## Create a User Defined Function in PHP

A user-defined function declaration starts with the word function:

### **Syntax**

function *functionName*() {  
*code to be executed*;  
}

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

**Tip:** Give the function a name that reflects what the function does!

In the example below, we create a function named "writeMsg()". The opening curly brace ( { ) indicates the beginning of the function code, and the closing curly brace ( } ) indicates the end of the function. The function outputs "Hello world!". To call the function, just write its name followed by brackets ():

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
function writeMsg() {  
  echo "Hello world!";  
}  
  
writeMsg(); // call the function  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function1)

ADVERTISEMENT

## PHP Function Arguments

Information can be passed to functions through arguments. An argument is just like a variable.

Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

The following example has a function with one argument ($fname). When the familyName() function is called, we also pass along a name (e.g. Jani), and the name is used inside the function, which outputs several different first names, but an equal last name:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function2)

The following example has a function with two arguments ($fname and $year):

### **Example**

<?php  
function familyName($fname, $year) {  
  echo "$fname Refsnes. Born in $year <br>";  
}  
  
familyName("Hege", "1975");  
familyName("Stale", "1978");  
familyName("Kai Jim", "1983");  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function3)

## PHP is a Loosely Typed Language

In the example above, notice that we did not have to tell PHP which data type the variable is.

PHP automatically associates a data type to the variable, depending on its value. Since the data types are not set in a strict sense, you can do things like adding a string to an integer without causing an error.

In PHP 7, type declarations were added. This gives us an option to specify the expected data type when declaring a function, and by adding the strict declaration, it will throw a "Fatal Error" if the data type mismatches.

In the following example we try to send both a number and a string to the function without using strict:

### **Example**

<?php  
function addNumbers(int $a, int $b) {  
  return $a + $b;  
}  
echo addNumbers(5, "5 days");  
// since strict is NOT enabled "5 days" is changed to int(5), and it will return 10  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_non-strict)

To specify strict we need to set declare(strict\_types=1);. This must be on the very first line of the PHP file.

In the following example we try to send both a number and a string to the function, but here we have added the strict declaration:

### **Example**

<?php declare(strict\_types=1); // strict requirement  
  
function addNumbers(int $a, int $b) {  
  return $a + $b;  
}  
echo addNumbers(5, "5 days");  
// since strict is enabled and "5 days" is not an integer, an error will be thrown  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_strict)

The strict declaration forces things to be used in the intended way.

## PHP Default Argument Value

The following example shows how to use a default parameter. If we call the function setHeight() without arguments it takes the default value as argument:

### **Example**

<?php declare(strict\_types=1); // strict requirement  
function setHeight(int $minheight = 50) {  
  echo "The height is : $minheight <br>";  
}  
  
setHeight(350);  
setHeight(); // will use the default value of 50  
setHeight(135);  
setHeight(80);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function4)

## PHP Functions - Returning values

To let a function return a value, use the return statement:

### **Example**

<?php declare(strict\_types=1); // strict requirement  
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>";  
echo "2 + 4 = " . sum(2, 4);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_function5)

## PHP Return Type Declarations

PHP 7 also supports Type Declarations for the return statement. Like with the type declaration for function arguments, by enabling the strict requirement, it will throw a "Fatal Error" on a type mismatch.

To declare a type for the function return, add a colon ( : ) and the type right before the opening curly ( { )bracket when declaring the function.

In the following example we specify the return type for the function:

### **Example**

<?php declare(strict\_types=1); // strict requirement  
function addNumbers(float $a, float $b) : float {  
  return $a + $b;  
}  
echo addNumbers(1.2, 5.2);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_return_strict)

You can specify a different return type, than the argument types, but make sure the return is the correct type:

### **Example**

<?php declare(strict\_types=1); // strict requirement  
function addNumbers(float $a, float $b) : int {  
  return (int)($a + $b);  
}  
echo addNumbers(1.2, 5.2);  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_return_strict2)

## Passing Arguments by Reference

In PHP, arguments are usually passed by value, which means that a copy of the value is used in the function and the variable that was passed into the function cannot be changed.

When a function argument is passed by reference, changes to the argument also change the variable that was passed in. To turn a function argument into a reference, the & operator is used:

### **Example**

Use a pass-by-reference argument to update a variable:

<?php  
function add\_five(&$value) {  
  $value += 5;  
}  
  
$num = 2;  
add\_five($num);  
echo $num;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_func_pass_ref)

# PHP Arrays

[❮ Previous](https://www.w3schools.com/php/php_functions.asp)[Next ❯](https://www.w3schools.com/php/php_arrays_indexed.asp)

An array stores multiple values in one single variable:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_num)

## What is an Array?

An array is a special variable, which can hold more than one value at a time.

If you have a list of items (a list of car names, for example), storing the cars in single variables could look like this:

$cars1 = "Volvo";  
$cars2 = "BMW";  
$cars3 = "Toyota";

However, what if you want to loop through the cars and find a specific one? And what if you had not 3 cars, but 300?

The solution is to create an array!

An array can hold many values under a single name, and you can access the values by referring to an index number.

## Create an Array in PHP

In PHP, the array() function is used to create an array:

array();

In PHP, there are three types of arrays:

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

ADVERTISEMENT

## Get The Length of an Array - The count() Function

The count() function is used to return the length (the number of elements) of an array:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_length)

## Complete PHP Array Reference

For a complete reference of all array functions, go to our complete [PHP Array Reference](https://www.w3schools.com/php/php_ref_array.asp).

The reference contains a brief description, and examples of use, for each function!

# PHP Indexed Arrays

[❮ Previous](https://www.w3schools.com/php/php_arrays.asp)[Next ❯](https://www.w3schools.com/php/php_arrays_associative.asp)

## PHP Indexed Arrays

There are two ways to create indexed arrays:

The index can be assigned automatically (index always starts at 0), like this:

$cars = array("Volvo", "BMW", "Toyota");

or the index can be assigned manually:

$cars[0] = "Volvo";  
$cars[1] = "BMW";  
$cars[2] = "Toyota";

The following example creates an indexed array named $cars, assigns three elements to it, and then prints a text containing the array values:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_num)

## Loop Through an Indexed Array

To loop through and print all the values of an indexed array, you could use a for loop, like this:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_num_loop)

ADVERTISEMENT

## Complete PHP Array Reference

For a complete reference of all array functions, go to our complete [PHP Array Reference](https://www.w3schools.com/php/php_ref_array.asp).

The reference contains a brief description, and examples of use, for each function!

# PHP Associative Arrays

[❮ Previous](https://www.w3schools.com/php/php_arrays_indexed.asp)[Next ❯](https://www.w3schools.com/php/php_arrays_multidimensional.asp)

## PHP Associative Arrays

Associative arrays are arrays that use named keys that you assign to them.

There are two ways to create an associative array:

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

or:

$age['Peter'] = "35";  
$age['Ben'] = "37";  
$age['Joe'] = "43";

The named keys can then be used in a script:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_assoc)

## Loop Through an Associative Array

To loop through and print all the values of an associative array, you could use a foreach loop, like this:

### **Example**

<?php  
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");  
  
foreach($age as $x => $x\_value) {  
  echo "Key=" . $x . ", Value=" . $x\_value;  
  echo "<br>";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_assoc_loop)

ADVERTISEMENT

## Complete PHP Array Reference

For a complete reference of all array functions, go to our complete [PHP Array Reference](https://www.w3schools.com/php/php_ref_array.asp).

The reference contains a brief description, and examples of use, for each function!

# PHP Multidimensional Arrays

[❮ Previous](https://www.w3schools.com/php/php_arrays_associative.asp)[Next ❯](https://www.w3schools.com/php/php_arrays_sort.asp)

In the previous pages, we have described arrays that are a single list of key/value pairs.

However, sometimes you want to store values with more than one key. For this, we have multidimensional arrays.

## PHP - Multidimensional Arrays

A multidimensional array is an array containing one or more arrays.

PHP supports multidimensional arrays that are two, three, four, five, or more levels deep. However, arrays more than three levels deep are hard to manage for most people.

**The dimension of an array indicates the number of indices you need to select an element.**

* For a two-dimensional array you need two indices to select an element
* For a three-dimensional array you need three indices to select an element

## PHP - Two-dimensional Arrays

A two-dimensional array is an array of arrays (a three-dimensional array is an array of arrays of arrays).

First, take a look at the following table:

|  |  |  |
| --- | --- | --- |
| **Name** | **Stock** | **Sold** |
| Volvo | 22 | 18 |
| BMW | 15 | 13 |
| Saab | 5 | 2 |
| Land Rover | 17 | 15 |

We can store the data from the table above in a two-dimensional array, like this:

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

Now the two-dimensional $cars array contains four arrays, and it has two indices: row and column.

To get access to the elements of the $cars array we must point to the two indices (row and column):

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
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>";  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_multi)

We can also put a for loop inside another for loop to get the elements of the $cars array (we still have to point to the two indices):

### **Example**

<?php  
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>";  
}  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_multi2)

## Complete PHP Array Reference

For a complete reference of all array functions, go to our complete [PHP Array Reference](https://www.w3schools.com/php/php_ref_array.asp).

The reference contains a brief description, and examples of use, for each function!

# PHP Sorting Arrays

[❮ Previous](https://www.w3schools.com/php/php_arrays_multidimensional.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals.asp)

The elements in an array can be sorted in alphabetical or numerical order, descending or ascending.

## PHP - Sort Functions For Arrays

In this chapter, we will go through the following PHP array sort functions:

* sort() - sort arrays in ascending order
* rsort() - sort arrays in descending order
* asort() - sort associative arrays in ascending order, according to the value
* ksort() - sort associative arrays in ascending order, according to the key
* arsort() - sort associative arrays in descending order, according to the value
* krsort() - sort associative arrays in descending order, according to the key

## Sort Array in Ascending Order - sort()

The following example sorts the elements of the $cars array in ascending alphabetical order:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_sort_alpha)

The following example sorts the elements of the $numbers array in ascending numerical order:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_sort_num)

ADVERTISEMENT

## Sort Array in Descending Order - rsort()

The following example sorts the elements of the $cars array in descending alphabetical order:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_rsort_alpha)

The following example sorts the elements of the $numbers array in descending numerical order:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_rsort_num)

## Sort Array (Ascending Order), According to Value - asort()

The following example sorts an associative array in ascending order, according to the value:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_asort)

## Sort Array (Ascending Order), According to Key - ksort()

The following example sorts an associative array in ascending order, according to the key:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_ksort)

## Sort Array (Descending Order), According to Value - arsort()

The following example sorts an associative array in descending order, according to the value:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_arsort)

## Sort Array (Descending Order), According to Key - krsort()

The following example sorts an associative array in descending order, according to the key:

### **Example**

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

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_array_krsort)

## Complete PHP Array Reference

For a complete reference of all array functions, go to our complete [PHP Array Reference](https://www.w3schools.com/php/php_ref_array.asp).

The reference contains a brief description, and examples of use, for each function!

PHP Global Variables - Superglobals

[❮ Previous](https://www.w3schools.com/php/php_arrays_sort.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals_globals.asp)

Superglobals were introduced in PHP 4.1.0, and are built-in variables that are always available in all scopes.

PHP Global Variables - Superglobals

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

The PHP superglobal variables are:

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

The next chapters will explain some of the superglobals, and the rest will be explained in later chapters.

# PHP Superglobal - $GLOBALS

[❮ Previous](https://www.w3schools.com/php/php_superglobals.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals_server.asp)

Super global variables are built-in variables that are always available in all scopes.

## PHP $GLOBALS

$GLOBALS is a PHP super global variable which is used to access global variables from anywhere in the PHP script (also from within functions or methods).

PHP stores all global variables in an array called $GLOBALS[*index*]. The index holds the name of the variable.

The example below shows how to use the super global variable $GLOBALS:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?php  
$x = 75;  
$y = 25;  
   
function addition() {  
  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];  
}  
   
addition();  
echo $z;  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_global_global)

In the example above, since z is a variable present within the $GLOBALS array, it is also accessible from outside the function!

# PHP Superglobal - $\_SERVER

[❮ Previous](https://www.w3schools.com/php/php_superglobals_globals.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals_request.asp)

Super global variables are built-in variables that are always available in all scopes.

## PHP $\_SERVER

$\_SERVER is a PHP super global variable which holds information about headers, paths, and script locations.

The example below shows how to use some of the elements in $\_SERVER:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<?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'];  
?>

[Try it Yourself »](https://tryphp.w3schools.com/showphp.php?filename=demo_global_server)

The following table lists the most important elements that can go inside $\_SERVER:

|  |  |
| --- | --- |
| **Element/Code** | **Description** |
| $\_SERVER['PHP\_SELF'] | Returns the filename of the currently executing script |
| $\_SERVER['GATEWAY\_INTERFACE'] | Returns the version of the Common Gateway Interface (CGI) the server is using |
| $\_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['SERVER\_SOFTWARE'] | Returns the server identification string (such as Apache/2.2.24) |
| $\_SERVER['SERVER\_PROTOCOL'] | Returns the name and revision of the information protocol (such as HTTP/1.1) |
| $\_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['HTTP\_HOST'] | Returns the Host header from the current request |
| $\_SERVER['HTTP\_REFERER'] | Returns the complete URL of the current page (not reliable because not all user-agents support it) |
| $\_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['REMOTE\_PORT'] | Returns the port being used on the user's machine to communicate with the web server |
| $\_SERVER['SCRIPT\_FILENAME'] | Returns the absolute pathname of the currently executing script |
| $\_SERVER['SERVER\_ADMIN'] | Returns the value given to the SERVER\_ADMIN directive in the web server configuration file (if your script runs on a virtual host, it will be the value defined for that virtual host) (such as someone@w3schools.com) |
| $\_SERVER['SERVER\_PORT'] | Returns the port on the server machine being used by the web server for communication (such as 80) |
| $\_SERVER['SERVER\_SIGNATURE'] | Returns the server version and virtual host name which are added to server-generated pages |
| $\_SERVER['PATH\_TRANSLATED'] | Returns the file system based path to the current script |
| $\_SERVER['SCRIPT\_NAME'] | Returns the path of the current script |
| $\_SERVER['SCRIPT\_URI'] | Returns the URI of the current page |

# PHP Superglobal - $\_REQUEST

[❮ Previous](https://www.w3schools.com/php/php_superglobals_server.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals_post.asp)

Super global variables are built-in variables that are always available in all scopes.

## PHP $\_REQUEST

PHP $\_REQUEST is a PHP super global variable which is used to collect data after submitting an HTML form.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to this file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_REQUEST to collect the value of the input field:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<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>

# PHP Superglobal - $\_POST

[❮ Previous](https://www.w3schools.com/php/php_superglobals_request.asp)[Next ❯](https://www.w3schools.com/php/php_superglobals_get.asp)

Super global variables are built-in variables that are always available in all scopes.

## PHP $\_POST

PHP $\_POST is a PHP super global variable which is used to collect form data after submitting an HTML form with method="post". $\_POST is also widely used to pass variables.

The example below shows a form with an input field and a submit button. When a user submits the data by clicking on "Submit", the form data is sent to the file specified in the action attribute of the <form> tag. In this example, we point to the file itself for processing form data. If you wish to use another PHP file to process form data, replace that with the filename of your choice. Then, we can use the super global variable $\_POST to collect the value of the input field:

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<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>

# PHP Superglobal - $\_GET

[❮ Previous](https://www.w3schools.com/php/php_superglobals_post.asp)[Next ❯](https://www.w3schools.com/php/php_regex.asp)

Super global variables are built-in variables that are always available in all scopes.

## PHP $\_GET

PHP $\_GET is a PHP super global variable which is used to collect form data after submitting an HTML form with method="get".

$\_GET can also collect data sent in the URL.

Assume we have an HTML page that contains a hyperlink with parameters:

<html>  
<body>  
  
<a href="test\_get.php?subject=PHP&web=W3schools.com">Test $GET</a>  
  
</body>  
</html>

When a user clicks on the link "Test $GET", the parameters "subject" and "web" are sent to "test\_get.php", and you can then access their values in "test\_get.php" with $\_GET.

The example below shows the code in "test\_get.php":

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

<html>  
<body>  
  
<?php  
echo "Study " . $\_GET['subject'] . " at " . $\_GET['web'];  
?>  
  
</body>  
</html>

# PHP Regular Expressions

[❮ Previous](https://www.w3schools.com/php/php_superglobals_get.asp)[Next ❯](https://www.w3schools.com/php/php_forms.asp)

## What is a Regular Expression?

A regular expression is a sequence of characters that forms a search pattern. When you search for data in a text, you can use this search pattern to describe what you are searching for.

A regular expression can be a single character, or a more complicated pattern.

Regular expressions can be used to perform all types of text search and text replace operations.

## Syntax

In PHP, regular expressions are strings composed of delimiters, a pattern and optional modifiers.

$exp = "/w3schools/i";

In the example above, / is the **delimiter**, w3schools is the **pattern**that is being searched for, and i is a **modifier**that makes the search case-insensitive.

The delimiter can be any character that is not a letter, number, backslash or space. The most common delimiter is the forward slash (/), but when your pattern contains forward slashes it is convenient to choose other delimiters such as # or ~.

## Regular Expression Functions

PHP provides a variety of functions that allow you to use regular expressions. The preg\_match(), preg\_match\_all() and preg\_replace() functions are some of the most commonly used ones:

|  |  |
| --- | --- |
| **Function** | **Description** |
| preg\_match() | Returns 1 if the pattern was found in the string and 0 if not |
| preg\_match\_all() | Returns the number of times the pattern was found in the string, which may also be 0 |
| preg\_replace() | Returns a new string where matched patterns have been replaced with another string |

## Using preg\_match()

The preg\_match() function will tell you whether a string contains matches of a pattern.

### **Example**[**Get your own PHP Server**](https://www.w3schools.com/spaces/)

Use a regular expression to do a case-insensitive search for "w3schools" in a string:

<?php  
$str = "Visit W3Schools";  
$pattern = "/w3schools/i";  
echo preg\_match($pattern, $str); // Outputs 1  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_regex_match)

## Using preg\_match\_all()

The preg\_match\_all() function will tell you how many matches were found for a pattern in a string.

### **Example**

Use a regular expression to do a case-insensitive count of the number of occurrences of "ain" in a string:

<?php  
$str = "The rain in SPAIN falls mainly on the plains.";  
$pattern = "/ain/i";  
echo preg\_match\_all($pattern, $str); // Outputs 4  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_regex_match_all)

## Using preg\_replace()

The preg\_replace() function will replace all of the matches of the pattern in a string with another string.

### **Example**

Use a case-insensitive regular expression to replace Microsoft with W3Schools in a string:

<?php  
$str = "Visit Microsoft!";  
$pattern = "/microsoft/i";  
echo preg\_replace($pattern, "W3Schools", $str); // Outputs "Visit W3Schools!"  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_regex_replace)

ADVERTISEMENT

## Regular Expression Modifiers

Modifiers can change how a search is performed.

|  |  |
| --- | --- |
| **Modifier** | **Description** |
| i | Performs a case-insensitive search |
| m | Performs a multiline search (patterns that search for the beginning or end of a string will match the beginning or end of each line) |
| u | Enables correct matching of UTF-8 encoded patterns |

## Regular Expression Patterns

Brackets are used to find a range of characters:

|  |  |
| --- | --- |
| **Expression** | **Description** |
| [abc] | Find one character from the options between the brackets |
| [^abc] | Find any character NOT between the brackets |
| [0-9] | Find one character from the range 0 to 9 |

## Metacharacters

Metacharacters are characters with a special meaning:

|  |  |
| --- | --- |
| **Metacharacter** | **Description** |
| | | Find a match for any one of the patterns separated by | as in: cat|dog|fish |
| . | Find just one instance of any character |
| ^ | Finds a match as the beginning of a string as in: ^Hello |
| $ | Finds a match at the end of the string as in: World$ |
| \d | Find a digit |
| \s | Find a whitespace character |
| \b | Find a match at the beginning of a word like this: \bWORD, or at the end of a word like this: WORD\b |
| \uxxxx | Find the Unicode character specified by the hexadecimal number xxxx |

## Quantifiers

Quantifiers define quantities:

|  |  |
| --- | --- |
| **Quantifier** | **Description** |
| n+ | Matches any string that contains at least one n |
| n\* | Matches any string that contains zero or more occurrences of n |
| n? | Matches any string that contains zero or one occurrences of n |
| n{x} | Matches any string that contains a sequence of *X* *n*'s |
| n{x,y} | Matches any string that contains a sequence of X to Y *n*'s |
| n{x,} | Matches any string that contains a sequence of at least X *n*'s |

**Note:** If your expression needs to search for one of the special characters you can use a backslash ( \ ) to escape them. For example, to search for one or more question marks you can use the following expression: $pattern = '/\?+/';

## Grouping

You can use parentheses ( ) to apply quantifiers to entire patterns. They also can be used to select parts of the pattern to be used as a match.

### **Example**

Use grouping to search for the word "banana" by looking for ba followed by two instances of na:

<?php  
$str = "Apples and bananas.";  
$pattern = "/ba(na){2}/i";  
echo preg\_match($pattern, $str); // Outputs 1  
?>

[Try it Yourself »](https://www.w3schools.com/php/phptryit.asp?filename=tryphp_regex_grouping)

## Complete RegExp Reference

For a complete reference, go to our [Complete PHP Regular Expression Reference](https://www.w3schools.com/php/php_ref_regex.asp).

The reference contains descriptions and examples of all Regular Expression functions.

Haut du formulaire