

PHP Tutorial

PHP is a powerful tool for making dynamic and interactive Web pages. PHP is a widely-used, free, and efficient alternative to competitors such as Microsoft's ASP. In this PHP tutorial from w3schools.com you will learn about PHP, and how to execute scripts on your server.

Code examples in this tutorial are color coded for the way DreamWeaver does its coloring for code hinting.



Introduction

PHP is a server-side scripting language.

What You Should Already Know

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

- » HTML/XHTML

If you want to study these subjects first, find the tutorials on w3cSchools.com.

What is PHP?

- » PHP stands for PHP: Hypertext Preprocessor
- » PHP is a server-side scripting language, like ASP
- » PHP scripts are executed on the server
- » PHP supports many databases (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC, etc.)
- » PHP is an open source software
- » PHP is free to download and use

What is a PHP File?

- » PHP files can contain text, HTML tags and scripts
- » PHP files are returned to the browser as plain HTML
- » PHP files have a file extension of ".php", ".php3", or ".phtml"

What is MySQL?

- » MySQL is a database server
- » MySQL is ideal for both small and large applications
- » MySQL supports standard SQL
- » MySQL compiles on a number of platforms
- » MySQL is free to download and use

PHP + MySQL

PHP combined with MySQL are cross-platform (you can develop in Windows and serve on a Unix platform)

Why PHP?

- » PHP runs on different platforms (Windows, Linux, Unix, etc.)
- » PHP is compatible with almost all servers used today (Apache, IIS, etc.)
- » PHP is FREE to download from the official PHP resource: www.php.net
- » PHP is easy to learn and runs efficiently on the server side

Where to Start?

To get access to a web server with PHP support, you can:

- » Find a web hosting plan with PHP and MySQL support
- » Or install Apache (or IIS) on your own server, install PHP, and MySQL

Syntax

PHP code is executed on the server, and the plain HTML result is sent to the browser.

Basic PHP Syntax

A PHP scripting block always starts with `<?php` and ends with `?>`. A PHP scripting block can be placed anywhere in the document.

On servers with shorthand support enabled you can start a scripting block with `<?` and end with `?>`.

For maximum compatibility, we recommend that you use the standard form (`<?php`) rather than the shorthand form.

```
<?php  
?>
```

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

Below, we have an example of a simple PHP script which sends the text "Hello World" to the browser:

```
<html>  
<body>  
  
<?php  
echo "Hello World";  
?>  
  
</body>  
</html>
```

Each code line in PHP must end with a semicolon. The semicolon is a separator and is used to distinguish one set of instructions from another.

There are two basic statements to output text with PHP: `echo` and `print`. In the example above we have used the `echo` statement to output the text "Hello World".

Note: The file must have a `.php` extension. If the file has a `.html` extension, the PHP code will not be executed.

Comments in PHP

In PHP, we use `//` to make a single-line comment or `/*` and `*/` to make a large comment block.

```
<html>
<body>

<?php

//This is a comment

/*
This is
a comment
block
*/

?>

</body>
</html>
```

Variables

A variable is used to store information.

Variables in PHP

Variables are used for storing a values, like text strings, numbers or arrays. When a variable is declared, it can be used over and over again in your script. All variables in PHP start with a \$ sign symbol.

The correct way of declaring a variable in PHP:

```
$var_name = value;
```

New PHP programmers often forget the \$ sign at the beginning of the variable. In that case it will not work. Let's try creating a variable containing a string, and a variable containing a number:

```
<?php  
  
$txt="Hello World!";  
  
$x=16;  
  
?>
```

PHP is a Loosely Typed Language

In PHP, a variable does not need to be declared before adding a value to it. In the example above, you see that you do not have to tell PHP which data type the variable is.

PHP automatically converts the variable to the correct data type, depending on its value. In a strongly typed programming language, you have to declare (define) the type and name of the variable before using it.

In PHP, the variable is declared automatically when you use it.

Naming Rules for Variables

- » A variable name must start with a letter or an underscore "_"
- » A variable name can only contain alpha-numeric characters and underscores (a-z, A-Z, 0-9, and _)
- » A variable name should not contain spaces. If a variable name is more than one word, it should be separated with an underscore (\$my_string), or with capitalization (\$myString)

String Variables

A string variable is used to store and manipulate text.

String Variables in PHP

String variables are used for values that contains characters. In this chapter we are going to look at the most common functions and operators used to manipulate strings in PHP. After we create a string we can manipulate it. A string can be used directly in a function or it can be stored in a variable.

Below, the PHP script assigns the text "Hello World" to a string variable called \$txt:

```
<?php  
  
$txt="Hello World";  
echo $txt;  
  
?>
```

The output of the code above will be:

```
Hello World
```

Now, lets try to use some different functions and operators to manipulate the string.

The Concatenation Operator

There is only one string operator in PHP. The concatenation operator (.) is used to put two string values together. To concatenate two string variables together, use the concatenation operator:

```
<?php  
  
$txt1 = "Hello World!";  
$txt2 = "What a nice day!";  
echo $txt1 . " " . $txt2;  
  
?>
```

The output of the code above will be:

```
Hello World! What a nice day!
```

If we look at the code above you see that we used the concatenation operator two times. This is because we had to insert a third string (a space character), to separate the two strings.

The strlen() function

The strlen() function is used to return the length of a string.

Let's find the length of a string:

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

The output of the code above will be:

```
12
```

The length of a string is often used in loops or other functions, when it is important to know when the string ends. (i.e. in a loop, we would want to stop the loop after the last character in the string).

The strpos() function

The strpos() function is used to search for character within a string. If a match is found, this function will return the position of the first match. If no match is found, it will return FALSE.

Let's see if we can find the string "world" in our string:

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

The output of the code above will be:

```
6
```

The position of the string "world" in our string is position 6. The reason that it is 6 (and not 7), is that the first position in the string is 0, and not 1.

Operators

Operators are used to operate on values.

PHP Operators

This section lists the different operators used in PHP.

Arithmetic Operators

Operator	Description	Example	Result
+	Addition	x=2 x+2	4
-	Subtraction	x=2 5-x	3
*	Multiplication	x=4 x*5	20
/	Division	15/5 5/2	3 2.5
%	Modulus (division remainder)	5%2 10%8 10%2	1 2 0
++	Increment	x=5 x++	x=6
--	Decrement	x=5 x--	x=4

Assignment Operators

Operator	Example	Is The Same As
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
=	x=y	x=x*y
/=	x/=y	x=x/y
.=	x.=y	x=x.y
%=	x%=y	x=x%y

Comparison Operators

Operator	Description	Example
<code>==</code>	is equal to	<code>5==8</code> returns false
<code>!=</code>	is not equal	<code>5!=8</code> returns true
<code><></code>	is not equal	<code>5<>8</code> returns true
<code>></code>	is greater than	<code>5>8</code> returns false
<code><</code>	is less than	<code>5<8</code> returns true
<code>>=</code>	is greater than or equal to	<code>5>=8</code> returns false
<code><=</code>	is less than or equal to	<code>5<=8</code> returns true

Logical Operators

Operator	Description	Example
<code>&&</code>	and	<code>x=6</code> <code>y=3</code> <code>(x < 10 && y > 1)</code> returns true
<code> </code>	or	<code>x=6</code> <code>y=3</code> <code>(x==5 y==5)</code> returns false
<code>!</code>	not	<code>x=6</code> <code>y=3</code> <code>!(x==y)</code> returns true

If...Else Statements

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

Conditional Statements

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

In PHP we have the following conditional statements:

- » if statement - use this statement to execute some code only if a specified condition is true
- » if...else statement - use this statement to execute some code if a condition is true and another code if the condition is false
- » if...elseif...else statement - use this statement to select one of several blocks of code to be executed
- » switch statement - use this statement to select one of many blocks of code to be executed

The if Statement

Use the if statement to execute some code only if a specified condition is true.

Syntax

if (condition) code to be executed if condition is true;

The following example will output "Have a nice weekend!" if the current day is Friday:

```
<html>
<body>

<?php

$d = date("D");
if ($d == "Fri") { echo "Have a nice weekend!"; }

?>

</body>
</html>
```

Notice that there is no ..else.. in this syntax. You tell the browser to execute some code only if the specified condition is true.

The if...else Statement

Use the if...else statement to execute some code if a condition is true and another code if a condition is false.

Syntax

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

Example

The following example will output "Have a nice weekend!" if the current day is Friday, otherwise it will output "Have a nice day!":

```
<html>  
<body>  
  
<?php  
  
$d = date("D");  
if ($d == "Fri") {  
  echo "Have a nice weekend!";  
}  
else {  
  echo "Have a nice day!";  
}  
  
?>  
  
</body>  
</html>
```

The if...elseif....else Statement

Use the if...elseif...else statement to select one of several blocks of code to be executed.

Syntax

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

Example

The following example will output "Have a nice weekend!" if the current day is Friday, and "Have a nice Sunday!" if the current day is Sunday. Otherwise it will output "Have a nice day!":

```
<html>  
<body>  
  
<?php  
  
$d = date("D");  
if ($d == "Fri") {  
  echo "Have a nice weekend!";  
}  
elseif ($d == "Sun") {  
  echo "Have a nice Sunday!";  
}  
else {  
  echo "Have a nice day!";  
}  
  
?>  
  
</body>  
</html>
```