Chapter 1: Introduction to PHP

What is PHP?

PHP is an acronym for (PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

- PHP scripts are executed on the server.

What is a PHP File?

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

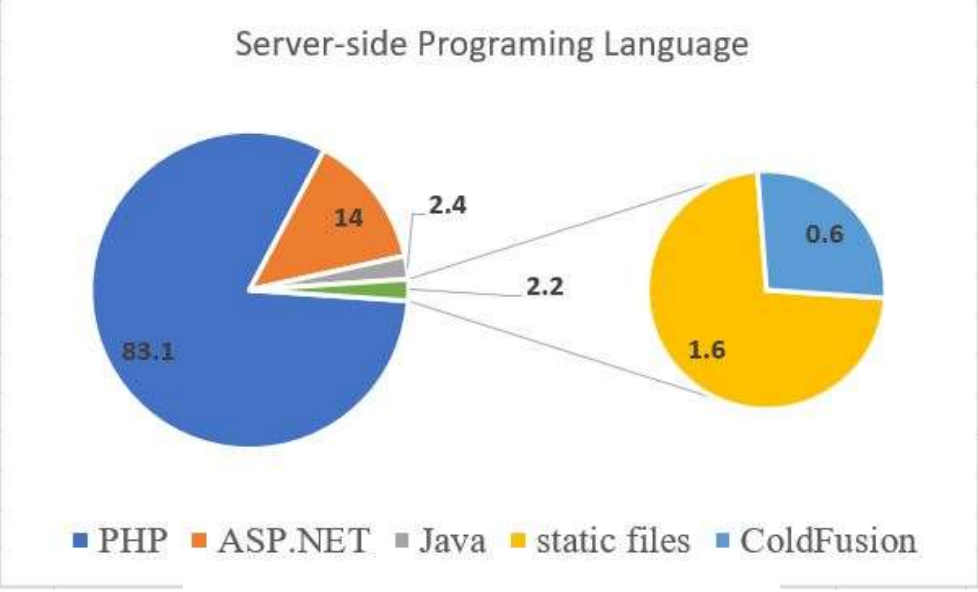
What Can PHP Do?

- 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, PDF files, and even Flash movies. You can also output any text, such as XHTML and XML.

Why PHP?

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



W3Techs https://w3techs.com/

Most popular server-side programming languages

© W3Techs.com	usage	change since 1 January 2018
1. PHP	83.1%	
2. ASP.NET	14.0%	-0.1%
3. Java	2.4%	-0.1%
4. static files	1.6%	+0.1%
5. ColdFusion	0.6%	

percentages of sites

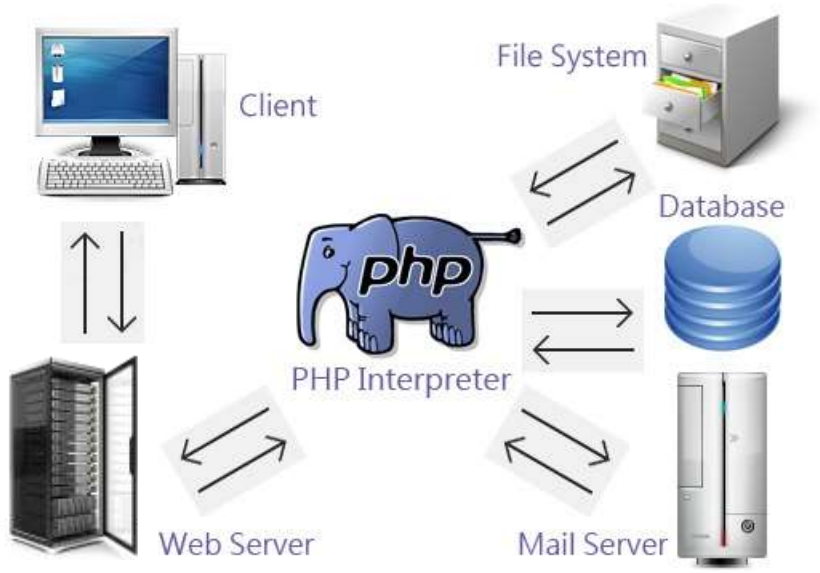
Fastest growing server-side programming languages since 1 January 2018

© W3Techs.com	sites
1. static files	311
2. JavaScript	21
3. ColdFusion	2

daily number of additional sites in the top 10 million

How Does PHP Work?

As Previously mentioned that PHP is a server-side programming language which means it runs in the server. PHP plays an intermediate role between a client and the date stored in the server and other servers.



What Do You Need to Start Using PHP?

- Install a web server
- Install PHP
- Install a database, such as MySQL

You can install a server on your personal computer such as *XAMPP* server from the link below:

https://www.apachefriends.org/index.html

A PHP version and MySQL come out of the box when you install *XAMPP* server. You can use other development server instead of XAMPP such as *WAMPSERVER* that you can download from here (http://www.wampserver.com/en/).

PHP Syntax:

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 **?>**

  Ex:

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

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

- **Comments in PHP**

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

  Ex:

```
// This is a single-line comment
# This is also a single-line comment
/*
This is a multiple-lines comment block
that spans over multiple
lines
*/
```

- **PHP Case Sensitivity**

  In PHP, all 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 legal (and equal):

Ex:

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

However; all variable names are case-sensitive.

In 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)

Ex:

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

PHP Variables:

- Start with a $
- Contain only letters, numbers, and the underscore
- The first character after the $ cannot be a number
- Are case-sensitive
- Use a consistent naming scheme!

Ex:

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

➢ 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

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

Ex:

```php
<?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>";
?>
```

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

Ex:

```php
<?php
function myTest() {
    $x = 5; // local scope
    echo "<p>Variable x inside function is: $x</p>";
}
myTest();
// using x outside the function will generate an error
echo "<p>Variable x outside function is: $x</p>";
?>
```

- **PHP The global Keyword**

  The global keyword is used to access a global variable from within a function.
  To do this, use the global keyword before the variables (inside the function):

Ex:

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

- **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:

  Ex:

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

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

Ex:

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

You can use either quotation mark type to create the string.

To use a variable within another string, you must use double quotation marks.

Ex:

```php
$first_name = 'Tobias';
$today = 'August 2, 2011';
$var = "Define \"platitude\", please.";
$var = 'Define "platitude", please.';
echo $first_name;
echo "Hello, $first_name";
```

➢ PHP Integer

- 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)

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

Ex:

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

- **PHP Float**

  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:

  Ex:

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

- **PHP Boolean**

  A Boolean represents two possible states: TRUE or FALSE.

  Ex:

```
$x = true;
$y = false;
```

- **PHP Array**

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

  Ex:

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

- **PHP Object**

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

  In PHP, an object must be explicitly declared.

  First we must declare a class of object. For this, we use the class keyword. A class is a structure that can contain properties and methods:

  Ex:

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

➢ 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:

Ex:

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

➢ 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