PHP Cheatsheet

This PHP cheatsheet provides a quick overview and revision of all concepts of PHP programming and is helpful for students and developers as well. If you are preparing for any exam or an interview, we recommend going through this cheatsheet to learn and enhance your PHP skills.

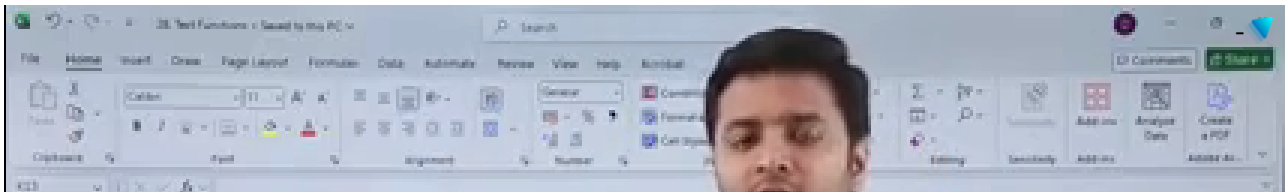
Script Syntax

Use the below syntax to write a PHP script −

```
<?php
   // Your PHP code here
?>
```

You can also use its short echo tag, which is −

```
<?= "Hello, World!" ?>
```



Printing "Hello, World!"

You can print "Hello, World!" in PHP using the PHP **echo** statement.

</> Open Compiler

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

Comments

PHP provides single and multiline comments. Where a single-line comment starts with **#** or **//** and a multiline comment is enclosed within **/*** and ***/**.

1. Single-line comment

```
// Write single-line comment here
# Write single-line comment here
```

2. Multi-line comment

```
/* You can write
multi-line comment like this*/
```

Variables

PHP variables store data and information. A variable can be declared using the dollar ($) sign followed by the variable name.

```
<?php
   $age = 21;
```

```
   $name = "Kelly Hu";
?>
```

Printing Variables

You can use the PHP echo statement to print variables.

</> Open Compiler

```
<?php
   $age = 21;
   $name = "Kelly Hu";
   echo "My name is $name";
   echo "<br>";
   echo "I am $age years old.";
?>
```

Printing Newline

You can print a newline using the echo statement by using the "\n" to print a newline in PHP. And, to display a newline in the browser, use the "
" tag.

1. Printing on Console

</> Open Compiler

```
<?php
   echo "Hello, World!\n";
   echo "This is on a new line.\n";
?>
```

2. Printing On Web Browser

</> Open Compiler

```
<?php
   echo "Hello, World!<br>";
```

```
   echo "This is on a new line.<br>";
?>
```

Printing Text

You can print text by using the following functions –

- echo
- print
- printf
- print_r

1. Printing Using echo

Use **echo** to print one or more strings.

</> Open Compiler

```
<?php
   echo "Hello, World!", " Hello PHP";
   $a = 10;
   $b = "Okay";
   echo "\nValue of a is $a and value of b is $b";
?>
```

2. Printing Using print

Use **print** to print one single string.

</> Open Compiler

```
<?php
   print "Hello, World!";
?>
```

3. Printing Using printf

Use **printf** to print formatted string.

</> Open Compiler

```
<?php
   printf("Hello, %s!", "World");
?>
```

4. Printing using print_r

Use print_r for debugging and displays human-readable information about variables, particularly arrays or objects.

</> Open Compiler

```
<?php
   $arr = array('Hyderabad', 'Bangalore', 'Mangalore');
   print_r($arr);
?>
```

Embedding HTML elements with Text

You can embed HTML elements with text using the echo and print function.

</> Open Compiler

```
<?php
   echo "<h1>Hello, World!</h1>";
   echo "<p>This is a paragraph.</p>";
?>
```

Embedding PHP in HTML

You can also embed PHP inside HTML elements.

</> Open Compiler

```
<html>
<body>
   <h1><?php echo "Welcome to My Website!"; ?></h1>
   <p><?php echo "This is a dynamic paragraph."; ?></p>
</body>
</html>
```

Combining PHP variables with HTML

You can include PHP variables within HTML elements as follows –

</> Open Compiler

```
<?php
   $name = "Kelly Hu";
   echo "<p>Hello, <em>$name</em>! Welcome to TutorialsPoint.</p>";
?>
```

Data Types

1. String

String data type represents a sequence of characters.

```
$str1 = "Hello, World!";
$str2 = 'PHP Data Types';
```

2. Integer

Integer data type represents non-decimal numbers between -2,147,483,648 and 2,147,483,647 (on 32-bit systems).

```
$int = 42;
```

3. Float (Double)

Float (double) data type represents numbers with a fractional part.

```
$float = 3.14;
```

4. Boolean

Boolean data type represents two possible values: TRUE or FALSE.

```
$is_admin = true;
$is_guest = false;
```

5. Array

PHP array type stores multiple values in a single variable.

```
// Indexed array
$cities = array("Hyderabad", "Bangalore", "Mangalore");
// Associative array
$person = array("name" => "Kelly Hu", "age" => 22);
```

6. Object

An array type is an instance of a class.

</> Open Compiler

```
<?php
   class Person {
      public $name;
      public function __construct($name) {
         $this->name = $name;
      }
      public function getName() {
         return $this->name;
      }
   }
   $p = new Person("Kelly Hu");
   echo $p->getName();  // Outputs: Kelly Hu
?>
```

7. NULL

The NULL type represents a variable with no value.

```
$var = null;
```

Strings

PHP strings are sequences of characters enclosed with either single quotes ('Hello') or double quotes ("Hello").

1. Creating and Printing String

You can create a string variable by assigning a string to a variable and print it by using the echo statement.

</> Open Compiler

```
<?php
   $str = "Kelly Hu";
   echo "Hello, $str!";
?>
```

2. String Concatenation

The dot (.) operator can be used to concatenate two strings.

</> Open Compiler

```
<?php
   $str = "Hello" . " " . "World!";
   echo $str;
?>
```

3. String Length

You can use the strlen() method to get the string length.

```
$len = strlen("Hello");  // Output: 5
```

4. String Functions

Some of the commonly used string functions that are used for various string operations are as follows –

Function Name	Use	Example
strlen()	Returns the length of a string.	`$str = "Hello World!";` `echo strlen($str);`
str_word_count()	Returns the number of words in a string.	`$str = "Hello World!";` `echo str_word_count($str);`
strrev()	Returns the reversed string.	`$str = "Hello";` `echo strrev($str);`
strpos()	Returns the position of the first occurrence of a substring.	`$str = "Hello World!";` `echo strpos($str, "World");`
str_replace()	Replaces some characters with others in a string.	`$str = "Hello World!";` `echo str_replace("World", "Harry", $str);`
strtolower()	Returns the lowercase converted string.	`$str = "HELLO WORLD!";` `echo strtolower($str);`

strtoupper()	Returns the uppercase converted string.	`$str = "hello world!";` `echo strtoupper($str);`
trim()	Removes whitespace or predefined characters from both ends.	`$str = " Hello World! ";` `echo trim($str);`
ltrim()	Removes whitespace or predefined characters from the left side.	`$str = " Hello World! ";` `echo ltrim($str);`
rtrim()	Removes whitespace or predefined characters from the right side.	`$str = " Hello World! ";` `echo rtrim($str);`
str_repeat()	Repeats a string a specified number of times.	`$str = "Hi! ";` `echo str_repeat($str, 3);`
substr()	Returns a part of a string.	`$str = "Hello World!";` `echo substr($str, 6, 5);`
ucfirst()	Returns the string having the first character of a string to uppercase.	`$str = "hello world!";` `echo ucfirst($str);`
ucwords()	Returns the string having the first character of each word in a string to uppercase.	`$str = "hello world!";` `echo ucwords($str);`

strcmp()	Compares two strings (case-sensitive).	`$str1 = "Hello"; $str2 = "hello"; echo strcmp($str1, $str2);`
strcasecmp()	Compares two strings (case-insensitive).	`$str1 = "Hello"; $str2 = "hello"; echo strcasecmp($str1, $str2);`
str_shuffle()	Randomly shuffles the characters of a string.	`$str = "Hello"; echo str_shuffle($str);`
number_format()	Formats a number with grouped thousands.	`$num = 1234567.89; echo number_format($num, 2);`

You can go through our reference page to read more about these methods: PHP String Functions Reference.

Escape Characters

PHP supports the following escape characters –

Escape Character	Description
\\	Backslash
\'	Single quote
\"	Double quote
\n	New line
\r	Carriage return
\t	Horizontal tab
\b	Backspace

Escape Character	Description
\f	Form feed
\v	Vertical tab
\0	Null character
\xHH	Hexadecimal value (HH is the hex code)
\u{HHHH}	Unicode character (PHP 7.2+)

Example of Escape Characters

</> Open Compiler

```
<?php
   echo "Hello\nWorld";
   echo "Hello\tWorld.\n";
   echo "This is single quote: It\'s Me.\n";
   echo "This is double quote: \"Hello, World!\"\n";
   echo "This is a backslash: \\\n";
?>
```

Operators

PHP operators are the symbols used to perform operations on variables and values. The following are the different types of operators –

List of All Operators

Operator Type	Operator	Description	Example
Arithmetic	+	Addition	5 + 3 (outputs 8)
	-	Subtraction	5 - 3 (outputs 2)
	*	Multiplication	5 * 3 (outputs 15)
	/	Division	5 / 2 (outputs 2.5)
	%	Modulus (remainder)	5 % 2 (outputs 1)

Operator Type	Operator	Description	Example
Assignment	=	Assigns value	$x = 5;
	+=	Adds and assigns	$x += 3; (same as $x = $x + 3;)
	-=	Subtracts and assigns	$x -= 2; (same as $x = $x - 2;)
	*=	Multiplies and assigns	$x *= 2; (same as $x = $x * 2;)
	/=	Divides and assigns	$x /= 2; (same as $x = $x / 2;)
	%=	Modulus and assigns	$x %= 3; (same as $x = $x % 3;)
Comparison	==	Equal (value)	($x == $y)
	===	Identical (value and type)	($x === $y)
	!=	Not equal	($x != $y)
	!==	Not identical (value or type)	($x !== $y)
	<	Less than	($x < $y)
	>	Greater than	($x > $y)
	<=	Less than or equal	($x <= $y)
	>=	Greater than or equal	($x >= $y)
Logical	&&	Logical AND	($x && $y)
	||	Logical OR	($x || $y)
	!	Logical NOT	!$x
String	.	Concatenation (combines two strings)	$str = "Hello" . " World";
Increment/Decrement	++	Increment (adds one)	++$x or $x++
	--	Decrement (subtracts one)	--$x or $x--

Example of PHP Operators

</> Open Compiler

```php
<?php
   // Arithmetic Operators
   $a = 10;
   $b = 3;
   echo $a + $b;
   echo $a % $b;

   // Assignment Operators
   $x = 5;
   $x += 2;
   echo $x;

   // Comparison Operators
   var_dump($x == 7);

   // Logical Operators
   if ($x > 5 && $x < 10) {
      echo "x is between 5 and 10.";
   }

   // String Operator
   $greeting = "Hello, " . "World!";
   echo $greeting;

   // Increment/Decrement Operators
   echo ++$x;
   echo $x--;
?>
```

Conditional Statements

PHP conditional statements are used to execute code based on conditions. The conditional statements are –

- if
- if else

- Multiple if else
- Nested if else
- switch

1. The if Statement

</> Open Compiler

```
<?php
   $x = 10;
   if ($x > 5) {
      echo "x is greater than 5";
   }
?>
```

2. The if else Statement

</> Open Compiler

```
<?php
   $x = 3;
   if ($x > 5) {
      echo "x is greater than 5";
   } else {
      echo "x is not greater than 5";
   }
?>
```

3. Multiple if else Statement

</> Open Compiler

```
<?php
   $x = 10;
   if ($x > 10) {
      echo "x is greater than 10";
```

```
   } elseif ($x == 10) {
      echo "x is equal to 10";
   } else {
      echo "x is less than 10";
   }
?>
```

4. Nested if else Statement

</> Open Compiler

```
<?php
   $x = 10;
   if ($x > 10) {
      echo "x is greater than 10";
   } elseif ($x == 10) {
      echo "x is equal to 10";
   } else {
      echo "x is less than 10";
   }
?>
```

5. The switch Statement

</> Open Compiler

```
<?php
   $day = 3;
   switch ($day) {
      case 1:
         echo "Monday";
         break;
      case 2:
         echo "Tuesday";
         break;
      case 3:
         echo "Wednesday";
         break;
```

```
      case 4:
         echo "Thursday";
         break;
      case 5:
         echo "Friday";
         break;
      case 6:
         echo "Saturday";
         break;
      case 7:
         echo "Sunday";
         break;
      default:
      echo "Invalid day";
   }
?>
```

Math Functions

PHP math functions are used for performing various mathematical operations. The following are the math functions –

Function	Example
abs()	echo abs(-5); // Outputs: 5
ceil()	echo ceil(4.3); // Outputs: 5
floor()	echo floor(4.7); // Outputs: 4
round()	echo round(4.5); // Outputs: 5
max()	echo max(1, 3, 2); // Outputs: 3
min()	echo min(1, 3, 2); // Outputs: 1
pow()	echo pow(2, 3); // Outputs: 8
sqrt()	echo sqrt(16); // Outputs: 4
rand()	echo rand(1, 100); // Outputs: random number
mt_rand()	echo mt_rand(1, 100); // Outputs: random number
sin()	echo sin(deg2rad(90)); // Outputs: 1

Function	Example
cos()	echo cos(deg2rad(180)); // Outputs: -1
tan()	echo tan(deg2rad(45)); // Outputs: 1
deg2rad()	echo deg2rad(180); // Outputs: 3.14159
rad2deg()	echo rad2deg(M_PI); // Outputs: 180

Constants

PHP constants are the variables with a fixed value. Constants are defined using **define()**.

</> Open Compiler

```
<?php
   define("SITE_NAME", "MyWebsite");
   echo SITE_NAME;
?>
```

Magic Constants

PHP magic constants are special constants with special meanings. The following are the magic constants –

Constant Name	Description
__LINE__	Current line number of the file.
__FILE__	Full path and filename of the file.
__DIR__	Directory of the file.
__FUNCTION__	Name of the function (case-sensitive).
__CLASS__	Name of the class (case-sensitive).
__METHOD__	Name of the method (case-sensitive).
__NAMESPACE__	Current namespace (empty if none).
__TRAIT__	Name of the trait (case-sensitive).

Example of Magic Constants

```
<?php
   echo "Current line: " . __LINE__ . "<\n";
   echo "Full file path: " . __FILE__ . "<\n";
   echo "Directory: " . __DIR__ . "<\n";

   function tpFunction() {
      echo "Function name: " . __FUNCTION__ . "<\n";
   }

   class tpClass {
      public function tpFun() {
         echo "Method name: " . __METHOD__ . "<\n";
      }
   }

   namespace MyNamespace;
   echo "Namespace: " . __NAMESPACE__ . "<\n";
?>
```

Loops

PHP loops repeat a block of code multiple times. The following are the types of loops –

- for
- foreach
- while
- do...while

1. The for Loop

</> Open Compiler

```
<?php
   echo "The for Loop:<\n";
   for ($i = 0; $i < 10; $i++) {
      echo "Iteration: $i<\n";
```

```
   }
?>
```

2. The foreach Loop

</> Open Compiler

```
<?php
   echo "The foreach Loop:<\n";
   $cities = ['Hyderabad', 'Bangalore', 'Mangalore'];
   foreach ($cities as $city) {
      echo "City: $city<\n";
   }
?>
```

3. The while Loop

</> Open Compiler

```
<?php
   echo "The while Loop:<\n";
   $count = 0;
   while ($count < 10) {
      echo "Count: $count<\n";
      $count++;
   }
?>
```

4. The do...while Loop

</> Open Compiler

```
<?php
   echo "The do...while Loop:<\n";
   $count = 0;
   do {
      echo "Count: $count<\n";
```

```
      $count++;
   } while ($count < 10);
?>
```

Functions

PHP functions are blocks of code that can be executed on demand. The following is an example of a simple function –

</> Open Compiler

```
<?php
   // Defining function
   function printName($name) {
      return "Hello, " . $name . "!";
   }

   // Calling function
   echo printName("Kelly Hu");
?>
```

Arrays

PHP arrays are used to store multiple values of the same or different types. The following are the different types of arrays –

- Indexed
- Associative
- Multidimensional

1. The Indexed Array

</> Open Compiler

```
<?php
   $cities = ["Hyderabad", "Bangalore", "Mangalore"];

   // Accessing elements
```

```
   echo $cities[0];
   echo $cities[1];
   echo $cities[2];
?>
```

2. The Associative Array

</> Open Compiler

```
<?php
   // Associative array with named keys
   $student = [
      "name" => "Kelly Hu",
      "age" => 21,
      "city" => "Brentwood"
   ];

   // Accessing elements by key
   echo $student["name"];
   echo $student["age"];
   echo $student["city"];
?>
```

Global Variables - Superglobals

PHP global variables (also known as superglobals) are built-in global arrays that are used to provide information or track variables. The superglobals are:

Superglobal	Description
$_GET	Collects data sent via URL parameters in a GET request.
$_POST	Collects data sent via HTTP POST method.
$_REQUEST	Collects data from $_GET, $_POST, and $_COOKIE.
$_SESSION	Stores data across pages for individual users.
$_COOKIE	Stores data on the clients browser as small text files.
$_FILES	Manages file uploads.

Superglobal	Description
$_ENV	Holds environment variables.
$_SERVER	Contains information about server and execution environment.
$_GLOBALS	Accesses all global variables in the PHP script.

Regular Expressions

PHP regular expressions are the patterns used for matching strings.

1. Validate an Email Address

</> Open Compiler

```
<?php
   $email = "myemail@domain.com";
   if (preg_match("/^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$/",
$email)) {
      echo "Valid Email.";
   } else {
      echo "Invalid Email";
   }
?>
```

2. Validate a Phone Number

</> Open Compiler

```
<?php
   $phone = "1234567890";
   if (preg_match("/^\d{10}$/", $phone)) {
      echo "It's a valid phone number.";
   } else {
      echo "It's not a valid phone number.";
   }
?>
```

3. Validate a URL

</> Open Compiler

```
<?php
   $url = "https://www.tutorialspoint.com";
   if (preg_match("/\b(?:https?|ftp):\/\/[a-zA-Z0-9-]+\.[a-zA-Z]{2,6}\b/",
$url)) {
      echo "It's a valid URL";
   } else {
      echo "It's not a valid URL";
   }
?>
```

Form Handling

PHP form handling is all about collecting and validating the user data via HTML forms. Here is an example –

```
<form method="post" action="form.php">
   Name: <input type="text" name="name">
   <input type="submit" value="Submit">
</form>

<?php
   // Check if form was submitted or not
   if ($_SERVER["REQUEST_METHOD"] == "POST") {
      $name = htmlspecialchars($_POST['name']);
      if (!empty($name)) {
         echo "Hello, " . $name . "!";
      } else {
         echo "Fill your name.";
      }
   }
?>
```

Date and Time

Use the **date()** function to get the date and time. Here is an example –

</> Open Compiler

```php
<?php
   // Get the current date and time
   echo "Current Date: " . date("Y-m-d") . "<\n";
   echo "Current Time: " . date("H:i:s") . "<\n";
   echo "Full Date and Time: " . date("Y-m-d H:i:s");
?>
```

Include Files

Use the include statement to include files within another PHP file.

```php
include 'header.php';
```

File Handling

PHP file handling is used for creating, reading, writing, and managing files.

1. Open and Close a File

```php
<?php
   $file = fopen("file1.txt", "w");
   fclose($file);
?>
```

2. Write a File

```php
<?php
   $file = fopen("file1.txt", "w");
   fwrite($file, "Some text to file.");
   fclose($file);
?>
```

3. Read from a File

```php
<?php
   $file = fopen("file1.txt", "r");
   $content = fread($file, filesize("file1.txt"));
   echo $content;
   fclose($file);
?>
```

4. Append to a File

```php
<?php
   $file = fopen("file1.txt", "a");
   fwrite($file, "\nSome extra text.");
   fclose($file);
?>
```

5. Delete a File

```php
<?php
   if (file_exists("file1.txt")) {
      unlink("file1.txt");
      echo "DONE.";
   } else {
      echo "File doesn't exist";
   }
?>
```

File Upload

PHP file upload allows you to upload files to the server.

```php
move_uploaded_file($_FILES["file"]["tmp_name"], "uploads/" . $_FILES["file"]
["name"]);
```

Cookies

PHP cookies store small pieces of data on the clients computer.

```
setcookie("user", "Kelly Hu", time() + (86400 * 30), "/");
```

Sessions

PHP sessions store data across multiple pages for a user.

```
session_start();
$_SESSION["username"] = "Kelly Hu";
```

Filters

PHP filters are used to validate and sanitize data.

```
$email = filter_var($email, FILTER_VALIDATE_EMAIL);
```

Docker Tutorial

Kubernetes Tutorial

DSA Tutorial

Spring Boot Tutorial

SDLC Tutorial

Unix Tutorial

CERTIFICATIONS

Business Analytics Certification

Java & Spring Boot Advanced Certification

Data Science Advanced Certification

Cloud Computing And DevOps

Advanced Certification In Business Analytics

Artificial Intelligence And Machine Learning

DevOps Certification

Game Development Certification

Front-End Developer Certification

AWS Certification Training

Python Programming Certification

COMPILERS & EDITORS

Online Java Compiler

Online Python Compiler

Online Go Compiler

Online C Compiler

Online C++ Compiler

Online C# Compiler

Online PHP Compiler

Online MATLAB Compiler

Online Bash Compiler

Online SQL Compiler

Online Html Editor

Tutorials Point is a leading Ed Tech company striving to provide the best learning material on technical and non-technical subjects.