PHP Chapter 2 : PHP Functions

Table of Contents

Procedural Programming

List of **instructions** being executed step by step
- **Variables**
- **Operators**
- **Control** statements
  - Making **decisions**
  - **Loops** for repetitive tasks

and **function** calls, with
- **Arguments** (passed data from the calling function process)
- Returned values, the **return** statement (function ends, and passes control back to the line from which it was called)

On-line documentation for the functions : https://www.php.net/manual/en/funcref.php

Function Reference

Organized by **categories**
- Date and Time Related Extensions — https://www.php.net/manual/en/ref.datetime.php
- Text Processing — https://www.php.net/manual/en/book.strings.php
- Variable and Type Related Extensions — https://www.php.net/manual/en/book.array.php
- …

Function Reference

Date and Time

https://www.php.net/manual/en/ref.datetime.php

Formatting a date, default current date or using a timestamp.

The **timestamp** is a temporal information, i.e. an elapsed time in sec for a given date and time of an event, since 1970-01-01 00:00:00 .

time(), maketime(), strtotime() allow you to calculate a timestamp.

date() returns a date as a String, according to its 1st argument, a user defined format (e.g. "*Y-m-d H:m:i*")

Character	Description
W (upper case)	Week number of the year
O (lower case)	Year according to the week number
N (upper case)	Day of the week (Monday 1 to Sunday 7)
Z (lower case)	Day of the year

Extracted from https://www.php.net/manual/en/function.date.php

```php
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Date</title>
<style>body {font-family:Verdana;font-size:12pt;}</style>
</head>
<body>
<?php
$currentTS=time();//current timestamp
echo "<p>The current date is ".date("Y-m-d",$currentTS)."
    </p>";
echo "<p>or by default ".date("Y-m-d H:i:s")."</p>";

$givenTS=mktime(0,0,0,12,31,2019);
echo "<p>Last day of year ".date("Y-m-d",$givenTS)."</p>";

$sentenceTS=strtotime('+1 week',$givenTS);
echo "<p>Next week ".date("Y-m-d",$sentenceTS)."</p>";
?>
</body>
</html>
```

The current date is 2019-09-10

or by default 2019-09-10 20:33:47

Last day of year 2019-12-31

Next week 2020-01-07

The first week of the year is the week containing the first Thursday

Checking week number for 2021-01-01 :

```
48 $day1stJanuaryTS=strtotime('2021-01-01');
49 $date1stJanuary=date('Y-m-d',$day1stJanuaryTS);
50 echo $date1stJanuary." => YearWeek using Y W format: ".date('YW',$day1stJanuaryTS)
       ."<br>";
51 echo $date1stJanuary." => YearWeek using o W format: ".date('oW',$day1stJanuaryTS)
       ."<br>";
```

2021-01-01 => YearWeek using Y W format: ~~202153~~

2021-01-01 => YearWeek using o W format: 202053

December 2020

WN	Mon	Tue	Wed	Thu	Fri	Sat	Sun
49	30	1	2	3	4	5	6
50	7	8	9	10	11	12	13
51	14	15	16	17	18	19	20
52	21	22	23	24	25	26	27
53	28	29	30	31	1	2	3

January 2021

WN	Mon	Tue	Wed	Thu	Fri	Sat	Sun
53	28	29	30	31	1	2	3
01	4	5	6	7	8	9	10
02	11	12	13	14	15	16	17
03	18	19	20	21	22	23	24
04	25	26	27	28	29	30	31

In order to take into account the time zone for a region of the globe, use *date_default_timezone_set()* function

```
10 $Paris='Europe/Paris';
11 $Baku='Asia/Baku';
12 date_default_timezone_set($Paris);
13 echo "<p>Current Date Time with $Paris Time Zone : ".date('Y-m-d H:i:s')."</p>";
14 date_default_timezone_set($Baku);
15 echo "<p>Current Date Time with $Baku Time Zone : ".date('Y-m-d H:i:s')."</p>";
```

Current Date Time with Europe/Paris Time Zone : 2019-09-11 10:43:43

Current Date Time with Asia/Baku Time Zone : 2019-09-11 12:43:43

checkdate() function controls the validity of a date, in the following form of 3 arguments month, day and year, submitted from a web form, for instance

```
55 var_dump(checkdate(11,18,2019));
56 echo "<hr>";
57 var_dump(checkdate(6,31,2020));
```

```
boolean true
---
boolean false
```

For **localisation** purpose, you sometime need to write a full date (day and month) in a given language :

date() function returns a string in English only!

Use *setlocal(LC_TIME, 'country_code')*
strftime() with an approriate string argument describing the date format expected

⇒ Search for the approriate country code, either for Windows, Linux or a remote server

http://www.loc.gov/standards/iso639-2/php/code_list.php

ISO 639-2 Code	ISO 639-1 Code	English name of Language	French name of Language	German name of Language
aze	az	Azerbaijani	azéri	Aserbeidschanisch
fre (B) fra (T)	fr	French	français	Französisch

```
10  //localhost
11  echo "Current date : ".date('l d F Y')."</p>";
12  echo "Current date (default English) : ".strftime('%A %d %B %Y')."<hr>";
13  setlocale(LC_TIME, "fr");
14  echo "Current date (French) : ".strftime('%A %d %B %Y')."<hr>";
15  setlocale(LC_TIME, "it");
16  echo "Current date (Italian) : ".utf8_encode(strftime('%A %d %B %Y'))."<hr>";
17  //remote server Alwaysdata
18  setlocale(LC_TIME, "az_AZ.UTF8");
19  echo "Current date (Azerbaijani) : ".strftime('%A %d %B %Y')."<hr>";
```

localhost

Current date : Wednesday 11 September 2019
Current date (default English) : Wednesday 11 September 2019
Current date (French) : mercredi 11 septembre 2019
Current date (Italian) : mercoledì 11 settembre 2019

remote server

Current date (Azerbaijani) : çərşənbə 11 sentyabr 2019

Exercise : free of charge loan repayment schedule

Data

- Purchase amount
- Deposit
- Loan amount
- Number of monthly payments
- Monthly payment
- Repayment date

Character strings

Many functions exist to transform, manipulate and analyze character strings.
Some functions will be detailed to understand how to use them.
Before using a new function, you must document it, understand how to use it, understand its arguments and its return value, test the examples, and finally you must make it your own, to integrate it into your script.

1) Online documentation

Text transform such as changing the string case. https://www.php.net/manual/en/function.strtoupper.php

2) Definition and description

(PHP 4, PHP 5, PHP 7)
strtoupper — Make a string uppercase

```
strtoupper ( string $string ) : string
```

Argument is a String

Returned value is a String

3) Arguments and returned value

Parameters

string
The input string.

Return Values

Returns the uppercased string.

4) Proposed examples

Examples

Example #1 strtoupper() example

```php
<?php
$str = "Mary Had A Little Lamb and She LOVED It So";
$str = strtoupper($str);
echo $str; // Prints MARY HAD A LITTLE LAMB AND SHE LOVED IT SO
?>
```

5) Other proposed functions

See Also

- strtolower() - Make a string lowercase
- ucfirst() - Make a string's first character uppercase
- ucwords() - Uppercase the first character of each word in a string
- mb_strtoupper() - Make a string uppercase

6) Try the functions

```php
<?php
$text="Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nam eget dolor et odio
    finibus tristique.";
$text1=strtoupper($text);
$text2=strtolower($text1);
$text3=ucwords($text2);

echo "<p>Text in uppercase:<br>$text1</p>";

echo "<p>Text in lowercase:<br>$text2</p>";

echo "<p>First letter of each word in uppercase:<br>$text3</p>";
```

Text in uppercase
LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISCING ELIT. NAM EGET DOLOR ET ODIO FINIBUS TRISTIQUE.

Text in lowercase
lorem ipsum dolor sit amet, consectetur adipiscing elit. nam eget dolor et odio finibus tristique.

First letter of each word in uppercase
Lorem Ipsum Dolor Sit Amet, Consectetur Adipiscing Elit. Nam Eget Dolor Et Odio Finibus Tristique.

7) Integrate it to your script

```php
setlocale(LC_TIME,'fr');
echo "<p>Aujourd'hui, nous sommes le ".strftime("%A %d %B")."</p>";
echo "<p>Aujourd'hui, nous sommes le ".ucwords(strftime("%A %d %B"))."</p>";
```

Aujourd'hui, nous sommes le samedi 14 septembre

Aujourd'hui, nous sommes le Samedi 14 Septembre

Some functions are now presented, in a given context, to suggest their future use (Non-exhaustive list).
Argument type (e.g. String, Integer, …) and returned value type (e.g. String, Integer, …) are shown.
Description and suggestion of use is proposed. Fully document these functions and try them.

returned value type & function(arguments)	Description and suggestion to use
int **strlen** (string *source*) mixed **count_chars** (string *string* [, int *mode*]) mixed **str_word_count** (string *string* [, int *format*])	Count the number of characters of a string. Perform statistic analysis of character distribution in the string. Mode or format may be used to filter returned values depending on frequency. ASCII character code may be returned
string **nl2br** (string *source*)	Strings may contain line feed or carriage return, when saved in a database table field. These can be converted to for an HTML display.
string **trim** (string *source* [, string *charlist*]) string **ltrim** (string *source* [, string *charlist*]) string **rtrim** (string *source* [, string *charlist*])	Remove left, right, or left and right characters, by default whitespace, or \t (tab), \n (line feed) \r (carriage return) if listed in the second argument. Useful to remove whitespace, if any, before writing a string in a table database.
string **strip_tags** (string *source* [, string *allowable_tags*])	When a HTML form is submitted, form field may contain HTML markup you want to remove, for security reason, for instance.
string **addslashes** (string *source*) string **stripslashes** (string *source*) string **addcslashes** (string *source*, string *charlist*) string **stripcslashes** (string *source*, string *charlist*)	Before writing string data into a database table field, single quotes ', double quotes ", and backslashes \ which can cause problems, can be escaped with a \ . The string charlist argument can specify which characters must be escaped. strip… functions remove the \ escape character.
string **number_format** (float *number* [, int *decimal_places*])	A simple pretty printing number, with, for instance 2 decimals only. Note that the returned value is a string, which can no longer be calculated.
string **md5** (string *source* [, bool *raw_output*]) string **sha1** (string *source* [, bool *raw_output*]) string **password_hash** (string *password*, int *algorithm* [, array *options*]) bool **password_verify** (string *password*, string *hash*)	md5() "Message Direct" gives a 32 character data hash, sha1() "Secure Hash Algorithm" gives a 40 character data hash, with the same algorithm. In a login / password database table, it is better to store hashed password. Use **password_** functions to handle with password.

returned value type & function(arguments)	Description and suggestion to use
string **substr** (string *source*, int *start* [, int *length*])	This built-in function is used to extract a part of a string. Without the *length* argument, the whole rest of the string is returned from the *start* position. A negative value for *start* returns the characters from the end of the string.
mixed **str_replace** (mixed *search*, mixed *replace*) mixed **str_ireplace** (mixed *search*, mixed *replace*)	mixed type means that the function works with string or array of strings. **i** stands for case insensitive.
int **strpos** (string *haystack*, string *needle*) int **stripos** (string *haystack*, string *needle*)	This function allows you to find the position of the first occurrence of a string in another string. It returns an integer value of the position of the first occurrence of the string. **i** stands for case insensitive (by default, the search is case sensitive)
string **strstr** (string *haystack*, string *needle*) string **stristr** (string *haystack*, string *needle*)	This function searches for the first occurrence of a string within another string and displays the rest of that string from the matching point.
string **chr** (int *ascii*) int **ord** (string *string*)	These functions are used to to convert a ASCII value to a character, and inversely.
string **htmlentities** (string *source*) string **html_entity_decode** (string *source*) string **htmlspecialchars** (string *source*) string **urlencode** (string *source*) string **urldecode** (string *source*)	Often, it is preferable to replace accented characters or reserved charcaters with their HTML entity. URL encoding, or percent encoding, onverts reserved, unsafe, and non-ASCII characters in URLs by a hexadecimal digits preceded by a percent sign (%), making the URL safe.

Many other functions exist, to be searched or discovered according to your needs…

Arrays

An array is a type of data structure which allows to store several values under a single variable, instead of creating a variable for each data value.
Array elements are indexed (array index, or **key**) either via a numeric index (integer), or a string index. We then refer to it as an associative array.
Multidimensional arrays are possible, where a given array element is itself a new array.

An array is created using an *array()* **function**. The first array index is 0. The index is between square brackets [ ]

```
9  $scores=array(12,9,13,14);
10 print_r($scores);
11 $scores[4]=10;
12 print_r($scores);
13 $scores[]=15;
14 print_r($scores);
```

```
Array ( [0] => 12 [1] => 9 [2] => 13 [3] => 14 )
Array ( [0] => 12 [1] => 9 [2] => 13 [3] => 14 [4] => 10 )
Array ( [0] => 12 [1] => 9 [2] => 13 [3] => 14 [4] => 10 [5] => 15 )
```

Automatic index incrementing

Loops can be used to access all array elements, either a *for* loop, or better, a *foreach* loop. In the case of a *for* loop, the *count()* function returns the number of elements.

```
18 $nbElements=count($scores);
19 for($i=0;$i<$nbElements;$i++){
20     echo "element $i = ".$scores[$i]."<br>";
21 }
22 foreach ($scores as $key => $value) {
23     echo "element $key = $value<br>";
24 }
```

```
element 0 = 12
element 1 = 9
element 2 = 13
element 3 = 14
element 4 = 10
element 5 = 15
```

An array element can be declared as an array, which gives rise to a multidimensional array. A string indexed array is called an associative array.

```
Array
(
    [Math] => Array
        (
            [0] => 12
            [1] => 11
            [2] => 15
        )

    [Physics] => Array
        (
        )

    [Web Programming] => Array
        (
            [0] => 14
            [1] => 16
            [2] => 15
            [3] => 12
        )

)
```

```
26 $scoreBySubject=array("Math"=>array(),"Physics
       "=>array(),"Web Programming"=>array());
27 $scoreBySubject["Math"]=array(12,11,15);
28 $scoreBySubject["Web Programming"]=[14,16,15,
       12];
29 echo "<pre>";
30 print_r($scoreBySubject);
31 echo "</pre>";
```

Some functions are now presented, in a given context, to suggest their future use (Non-exhaustive list) ⇒ special array functions can be used to manipulate your data.
Argument type (e.g. String, Integer, …) and returned value type (e.g. String, Integer, …) are shown.
Description and suggestion of use is proposed. Fully document these functions and try them.

returned value type & function(arguments)	Description and suggestion to use
array **array** ([mixed ...])	Assign to your variable either the *array()* function, or the [ ] operator (short array syntax) to create an array.
int **count** (array *input*)	Returns the number of elements, usually the maximum index value +1. 1st element has an index of 0
bool **print_r** (array input) void **var_dump** (array input)	An easy way to display the array content, for debugging or during development stage.
bool **asort** (array *input*) bool **ksort** (array *input*) bool **arsort** (array *input*) bool **krsort** (array *input*)	These functions allow ordering data in an alphabetical, numerical order, in increasing or decreasing sequence, by sorting on values, or on keys. A second argument specifies how items should be compared : numerically SORT_NUMERIC, as strings SORT_STRING, …
array **array_diff** (array *array1*, array *array2* [, ...]) array **array_intersect** (array *array1*, array *array2* [, ...]) array **array_merge** (array *array1*, array *array2* [, ...])	*array_diff()* returns a new array containing all the values of array1 that do not exist in array2, *array_intersect()* returns a new array containing all the values of array1 that do exist in array2, *array_merge()* combines the two arrays.
array **array_unique** (array *input*)	This function allows you to delete duplicate data before performing the merge and intersection functions.
int **extract** (array *source*)	This function converts an associative array into individual variables, where the key (a string) becomes the name of the variable. **Be careful with security risks if you use this function on a $_POST array of data received from a form!!**

returned value type & function(arguments)	Description and suggestion to use
bool **in_array** (mixed *needle*, array *haystack*)	This function checks whether an array element exists.
bool **in_array** (mixed *needle*, array *haystack*) bool **array_key_exists** (mixed *needle*, array *haystack*)	Checking whether an element exists in an array, or a key exists. The first argument is the value searched (needle), the second is the array (haystack).

Many other functions exist, to be searched or discovered according to your needs…

Some usefull built-in functions

Some basic functions are widely used with single variables and arrays.

- *isset()* checks if a variable is set with a value.
- *empty()* checks if a variable is empty. A variable is not empty only if it contains a *string* value (but not *blank* value), a boolean value *true*, or a *numerical* value (but not 0).
- is_null() checks if the value of the variable is set to NULL (NULL value are often seen as database field values).

$variable Function	Not defined	=""	="PHP"	=NULL	=true	=false	=0	=1
isset()	false	true	true	false	true	true	true	true
empty()	true	true	false	true	false	true	true	false
is_null()	-	false	false	true	false	false	false	false

- *array_key_exists()* checks if a key exists in an array.

- *unset()* removes an existing variable entirely, so that *isset()* will return false.

- *print_r()* and *var_dump()* are often used to display an array content, for **debugging** or during development stage. The <pre> HTML element allows to keep the *print_r()* preformatted display.

```
29 print_r($scoreBySubject);
```

```
Array ( [Math] => Array ( [0] => 12 [1] => 11 [2] => 15 ) [Physics] => Array ( ) [Web Programming] => Array ( [0] => 14 [1] => 16 [2] => 15 [3] => 12 ) )
```

```
32 echo "</pre>";
33 extract($scoreBySubject);
34 print_r($Math);
```

```
Array
(
    [Math] => Array
        (
            [0] => 12
            [1] => 11
            [2] => 15
        )

    [Physics] => Array
        (
        )

    [Web Programming] => Array
        (
            [0] => 14
            [1] => 16
            [2] => 15
            [3] => 12
        )

)
```

Mathematical functions : PHP provides many predefined math constants and functions. All you have to do is search for them and document them, try them out,.....

Rounding and **number formatting** for display.
Use the functions *ceil()*, *floor()*, *round()* and *number_format()*.

```
Ceil of 5.625 gives 6
Floor of 5.625 gives 5
Round of 5.625 gives 6
The formatting of 5.625 with 2 digits after the decimal point displays 5.63
```

```
Ceil of 5.224 gives 6
Floor of 5.224 gives 5
Round of 5.224 gives 5
The formatting of 5.224 with 2 digits after the decimal point displays 5.22
```

```
9  $value=5.625;
10 echo "Ceil of $value gives ".ceil($value)."<
       br>";
11 echo "Floor of $value gives ".floor($value)."
       <br>";
12 echo "Round of $value gives ".round($value,0)
       ."<br>";
13 echo "The formatting of $value with 2 digits
       after the decimal point displays ".
       number_format($value,2)."<br>";
```

```
15 $value=5.224;
16 echo "Ceil of $value gives ".ceil($value)."<
       br>";
17 echo "Floor of $value gives ".floor($value)."
       <br>";
18 echo "Round of $value gives ".round($value,0)
       ."<br>";
19 echo "The formatting of $value with 2 digits
       after the decimal point displays ".
       number_format($value,2)."<br>";
```

For **trigonometrical computing**, use *sin()*, *cos()*,… , *deg2rad()*,… and constants **M_PI**, **M_PI_2**, …
For a full list, see https://www.php.net/manual/en/ref.math.php

User Defined Functions

Tips : choose a name function that reflects what it does! The function name begins with a letter or an underscore.

In addition to the many built-in functions that already exist, you can design your own function. A user defined function is a block of statements that can be run repeatedly.

Anatomy and Syntax of a function