**TUGAS MATERI ARRAY**

**FUNCTION PHP YANG BERKAITAN DENGAN ARRAY**



**Adam Arthur Faizal**

**M3119001**

**TI A**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**SEKOLAH VOKASI**

**UNIVERSITAS SEBELAS MARET**

**SURAKARTA**

**2020**

# 10 built-in function PHP yang berkaitan dengan array & contoh script nya

# 1. array\_count\_values

(PHP 4, PHP 5, PHP 7)

array\_count\_values — Counts all the values of an array

### Description

array\_count\_values ( array $array ) : array

array\_count\_values() returns an array using the values of array as keys and their frequency in array as values.

### Parameters

*array*

The array of values to count

### Return Values

Returns an associative array of values from array as keys and their count as value.

### Errors/Exceptions

Throws E\_WARNING for every element which is not [string](https://www.php.net/manual/en/language.types.string.php) or [integer](https://www.php.net/manual/en/language.types.integer.php).

### Examples

Example #1 array\_count\_values() example

<?php  
$array = array(1, "hello", 1, "world", "hello");  
print\_r(array\_count\_values($array));  
?>

The above example will output:

Array

(

[1] => 2

[hello] => 2

[world] => 1

)

# 2. array\_fill

(PHP 4 >= 4.2.0, PHP 5, PHP 7)

array\_fill — Fill an array with values

### Description

array\_fill ( int *$start\_index*, int *$num*, [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed) *$value* ) : array

Fills an array with num entries of the value of the value parameter, keys starting at the start\_index parameter.

### Parameters

*start\_index*

The first index of the returned array.

If start\_index is negative, the first index of the returned array will be start\_index and the following indices will start from zero (see [example](https://www.php.net/manual/en/function.array-fill.php" \l "function.array-fill.example.basic)).

*num*

Number of elements to insert. Must be greater than or equal to zero.

*value*

Value to use for filling

### Return Values

Returns the filled array

### Errors/Exceptions

Throws a E\_WARNING if num is less than zero.

### Examples

Example #1 array\_fill() example

<?php  
$a = array\_fill(5, 6, 'banana');  
$b = array\_fill(-2, 4, 'pear');  
print\_r($a);  
print\_r($b);  
?>

The above example will output:

Array

(

[5] => banana

[6] => banana

[7] => banana

[8] => banana

[9] => banana

[10] => banana

)

Array

(

[-2] => pear

[0] => pear

[1] => pear

[2] => pear

)

# 3. array\_filter

(PHP 4 >= 4.0.6, PHP 5, PHP 7)

array\_filter — Filters elements of an array using a callback function

### Description

array\_filter ( array *$array*, [callable](https://www.php.net/manual/en/language.types.callable.php) *$callback*, int *$flag* = 0 ) : array

Iterates over each value in the array passing them to the callback function. If the callback function returns TRUE, the current value from array is returned into the result [array](https://www.php.net/manual/en/language.types.array.php).

Array keys are preserved, and may result in gaps if the array was indexed. The result [array](https://www.php.net/manual/en/language.types.array.php) can be reindexed using the [array\_values()](https://www.php.net/manual/en/function.array-values.php) function.

### Parameters

*array*

The array to iterate over

*callback*

The callback function to use

If no callback is supplied, all empty entries of array will be removed. See [empty()](https://www.php.net/manual/en/function.empty.php) for how PHP defines empty in this case.

*flag*

Flag determining what arguments are sent to callback:

* ARRAY\_FILTER\_USE\_KEY - pass key as the only argument to callback instead of the value
* ARRAY\_FILTER\_USE\_BOTH - pass both value and key as arguments to callback instead of the value

Default is 0 which will pass value as the only argument to callback instead.

### Return Values

Returns the filtered array.

### Examples

Example #1 array\_filter() example

<?php  
function odd($var)  
{  
 // returns whether the input integer is odd  
 return $var & 1;  
}  
  
function even($var)  
{  
    // returns whether the input integer is even  
    return !($var & 1);  
}  
  
$array1 = ['a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' => 5];  
$array2 = [6, 7, 8, 9, 10, 11, 12];  
  
echo "Odd :\n";  
print\_r(array\_filter($array1, "odd"));  
echo "Even:\n";  
print\_r(array\_filter($array2, "even"));  
?>

The above example will output:

Odd :

Array

(

[a] => 1

[c] => 3

[e] => 5

)

Even:

Array

(

[0] => 6

[2] => 8

[4] => 10

[6] => 12

)

Example #2 array\_filter() without callback

<?php  
  
$entry = [  
    0 => 'foo',  
    1 => false,  
    2 => -1,  
    3 => null,  
    4 => '',  
    5 => '0',  
    6 => 0,  
];  
  
print\_r(array\_filter($entry));  
?>

The above example will output:

Array

(

[0] => foo

[2] => -1

)

Example #3 array\_filter() with flag

<?php  
  
$arr = ['a' => 1, 'b' => 2, 'c' => 3, 'd' => 4];  
  
var\_dump(array\_filter($arr, function($k) {  
    return $k == 'b';  
}, ARRAY\_FILTER\_USE\_KEY));  
  
var\_dump(array\_filter($arr, function($v, $k) {  
    return $k == 'b' || $v == 4;  
}, ARRAY\_FILTER\_USE\_BOTH));  
?>

The above example will output:

array(1) {

["b"]=>

int(2)

}

array(2) {

["b"]=>

int(2)

["d"]=>

int(4)

}

# 4. array\_replace

(PHP 5 >= 5.3.0, PHP 7)

array\_replace — Replaces elements from passed arrays into the first array

### Description

array\_replace ( array $array1 [, array $... ] ) : array

array\_replace() replaces the values of array1 with values having the same keys in each of the following arrays. If a key from the first array exists in the second array, its value will be replaced by the value from the second array. If the key exists in the second array, and not the first, it will be created in the first array. If a key only exists in the first array, it will be left as is. If several arrays are passed for replacement, they will be processed in order, the later arrays overwriting the previous values.

array\_replace() is not recursive : it will replace values in the first array by whatever type is in the second array.

### Parameters

array1

The array in which elements are replaced.

...

Arrays from which elements will be extracted. Values from later arrays overwrite the previous values.

### Return Values

Returns an [array](https://www.php.net/manual/en/language.types.array.php), or NULL if an error occurs.

### Examples

Example #1 array\_replace() example

<?php  
$base = array("orange", "banana", "apple", "raspberry");  
$replacements = array(0 => "pineapple", 4 => "cherry");  
$replacements2 = array(0 => "grape");  
  
$basket = array\_replace($base, $replacements, $replacements2);  
print\_r($basket);  
?>

The above example will output:

Array

(

[0] => grape

[1] => banana

[2] => apple

[3] => raspberry

[4] => cherry

)

# 5. array\_reverse

(PHP 4, PHP 5, PHP 7)

array\_reverse — Return an array with elements in reverse order

### Description

array\_reverse ( array $array [, bool $preserve\_keys = FALSE ] ) : array

Takes an input array and returns a new array with the order of the elements reversed.

### Parameters

array

The input array.

preserve\_keys

If set to TRUE numeric keys are preserved. Non-numeric keys are not affected by this setting and will always be preserved.

### Return Values

Returns the reversed array.

### Examples

Example #1 array\_reverse() example

<?php  
$input  = array("php", 4.0, array("green", "red"));  
$reversed = array\_reverse($input);  
$preserved = array\_reverse($input, true);  
  
print\_r($input);  
print\_r($reversed);  
print\_r($preserved);  
?>

The above example will output:

Array

(

[0] => php

[1] => 4

[2] => Array

(

[0] => green

[1] => red

)

)

Array

(

[0] => Array

(

[0] => green

[1] => red

)

[1] => 4

[2] => php

)

Array

(

[2] => Array

(

[0] => green

[1] => red

)

[1] => 4

[0] => php

)

# 6. array\_search

(PHP 4 >= 4.0.5, PHP 5, PHP 7)

array\_search — Searches the array for a given value and returns the first corresponding key if successful

### Description

array\_search ( [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed) $needle , array $haystack [, bool $strict = FALSE ] ) : [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed)

Searches for needle in haystack.

### Parameters

needle

The searched value.

Note:

If needle is a string, the comparison is done in a case-sensitive manner.

haystack

The array.

strict

If the third parameter strict is set to TRUE then the array\_search() function will search for identical elements in the haystack. This means it will also perform a [strict type comparison](https://www.php.net/manual/en/language.types.php) of the needle in the haystack, and objects must be the same instance.

### Return Values

Returns the key for needle if it is found in the array, FALSE otherwise.

If needle is found in haystack more than once, the first matching key is returned. To return the keys for all matching values, use [array\_keys()](https://www.php.net/manual/en/function.array-keys.php) with the optional search\_value parameter instead.

Warning

This function may return Boolean FALSE, but may also return a non-Boolean value which evaluates to FALSE. Please read the section on [Booleans](https://www.php.net/manual/en/language.types.boolean.php) for more information. Use [the === operator](https://www.php.net/manual/en/language.operators.comparison.php) for testing the return value of this function.

### Examples

Example #1 array\_search() example

<?php  
$array = array(0 => 'blue', 1 => 'red', 2 => 'green', 3 => 'red');  
  
$key = array\_search('green', $array); // $key = 2;  
$key = array\_search('red', $array);   // $key = 1;  
?>

# 7. count

(PHP 4, PHP 5, PHP 7)

count — Count all elements in an array, or something in an object

### Description

count ( [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed) $array\_or\_countable [, int $mode = COUNT\_NORMAL ] ) : int

Counts all elements in an array, or something in an object.

For objects, if you have [SPL](https://www.php.net/manual/en/ref.spl.php) installed, you can hook into count() by implementing interface [Countable](https://www.php.net/manual/en/class.countable.php). The interface has exactly one method, [Countable::count()](https://www.php.net/manual/en/countable.count.php), which returns the return value for the count() function.

Please see the [Array](https://www.php.net/manual/en/language.types.array.php) section of the manual for a detailed explanation of how arrays are implemented and used in PHP.

### Parameters

array\_or\_countable

An array or [Countable](https://www.php.net/manual/en/class.countable.php) object.

mode

If the optional mode parameter is set to COUNT\_RECURSIVE (or 1), count() will recursively count the array. This is particularly useful for counting all the elements of a multidimensional array.

Caution

count() can detect recursion to avoid an infinite loop, but will emit an E\_WARNING every time it does (in case the array contains itself more than once) and return a count higher than may be expected.

### Return Values

Returns the number of elements in array\_or\_countable. When the parameter is neither an array nor an object with implemented [Countable](https://www.php.net/manual/en/class.countable.php) interface, 1 will be returned. There is one exception, if array\_or\_countable is NULL, 0 will be returned.

### Examples

Example #1 count() example

<?php  
$a[0] = 1;  
$a[1] = 3;  
$a[2] = 5;  
var\_dump(count($a));  
  
$b[0]  = 7;  
$b[5]  = 9;  
$b[10] = 11;  
var\_dump(count($b));  
  
var\_dump(count(null));  
  
var\_dump(count(false));  
?>

The above example will output:

int(3)

int(3)

Warning: count(): Parameter must be an array or an object that implements Countable in … on line 12 // as of PHP 7.2

int(0)

Warning: count(): Parameter must be an array or an object that implements Countable in … on line 14 // as of PHP 7.2

int(1)

Example #2 Recursive count() example

<?php  
$food = array('fruits' => array('orange', 'banana', 'apple'),  
              'veggie' => array('carrot', 'collard', 'pea'));  
  
// recursive count  
echo count($food, COUNT\_RECURSIVE); // output 8  
  
// normal count  
echo count($food); // output 2  
  
?>

# 8. end

(PHP 4, PHP 5, PHP 7)

end — Set the internal pointer of an array to its last element

### Description

end ( array &$array ) : [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed)

end() advances array's internal pointer to the last element, and returns its value.

### Parameters

array

The array. This array is passed by reference because it is modified by the function. This means you must pass it a real variable and not a function returning an array because only actual variables may be passed by reference.

### Return Values

Returns the value of the last element or FALSE for empty array.

### Examples

Example #1 end() example

<?php  
  
$fruits = array('apple', 'banana', 'cranberry');  
echo end($fruits); // cranberry  
  
?>

# 9. range

(PHP 4, PHP 5, PHP 7)

range — Create an array containing a range of elements

### Description

range ( [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed) $start , [mixed](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.mixed) $end [, [number](https://www.php.net/manual/en/language.pseudo-types.php" \l "language.types.number) $step = 1 ] ) : array

Create an array containing a range of elements.

### Parameters

start

First value of the sequence.

end

The sequence is ended upon reaching the end value.

step

If a step value is given, it will be used as the increment between elements in the sequence. step should be given as a positive number. If not specified, step will default to 1.

### Return Values

Returns an array of elements from start to end, inclusive.

### Examples

Example #1 range() examples

<?php  
// array(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)  
foreach (range(0, 12) as $number) {  
    echo $number;  
}  
  
// The step parameter  
// array(0, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100)  
foreach (range(0, 100, 10) as $number) {  
    echo $number;  
}  
  
// Usage of character sequences  
// array('a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i');  
foreach (range('a', 'i') as $letter) {  
    echo $letter;  
}  
// array('c', 'b', 'a');  
foreach (range('c', 'a') as $letter) {  
    echo $letter;  
}  
?>

# 10. sizeof

(PHP 4, PHP 5, PHP 7)

sizeof — Alias of [count()](https://www.php.net/manual/en/function.count.php)

### Description

This function is an alias of: [count()](https://www.php.net/manual/en/function.count.php).

# DAFTAR PUSTAKA

PHP. PHP Manual. <https://www.php.net/manual/en/>. Tanggal diakses: 13 Oktober 2020

PHP. Array Functions. <https://www.php.net/manual/en/ref.array.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Count Values. <https://www.php.net/manual/en/function.array-count-values.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Fill. <https://www.php.net/manual/en/function.array-fill.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Filter. <https://www.php.net/manual/en/function.array-filter.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Keys. <https://www.php.net/manual/en/function.array-keys.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Replace. <https://www.php.net/manual/en/function.array-reverse.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Reverse. <https://www.php.net/manual/en/function.array-reverse.php>. Tanggal diakses: 13 Oktober 2020

PHP. Array Search. <https://www.php.net/manual/en/function.array-search.php>. Tanggal diakses: 13 Oktober 2020

PHP. Count. <https://www.php.net/manual/en/function.count.php>. Tanggal diakses; 13 Oktober 2020

PHP. End. <https://www.php.net/manual/en/function.end.php>. Tanggal diakses: 13 Oktober 2020

PHP. Range. <https://www.php.net/manual/en/function.range.php>. Tanggal diakses: 13 Oktober 2020

PHP. Sizeof. <https://www.php.net/manual/en/function.sizeof.php>. Tanggal diakses: 13 Oktober 2020