php

- Downloads
- Documentation
- Get Involved
- Help

Search

Keyboard Shortcuts

?
: This help

j
: Next menu item

k
: Previous menu item

g p
: Previous man page

g n
: Next man page

G
: Scroll to bottom

g g
: Scroll to top

g h
: Goto homepage

g s

Goto search
(current page)
/
Focus search box

소개 ≫
≪ Variable and Type Related Extensions

- PHP 매뉴얼
- 함수 레퍼런스
- Variable and Type Related Extensions

Change language: Other

Edit Report a Bug

배열 ¶

- array_multisort — 여러 배열이나 다차원 배열 정렬
- array_pad — 지정한 길이만큼 특정 값으로 배열 채우기
- array_pop — 배열의 마지막 원소 빼내기
- array_product — Calculate the product of values in an array
- array_push — 배열의 끝에 하나 이상의 원소를 넣는다
- array_rand — 배열에서 하나 이상의 임의 원소를 가져옴
- array_reduce — 콜백 함수를 사용하여 배열을 반복적으로 단일 값으로 축소
- array_replace_recursive — Replaces elements from passed arrays into the first array recursively
- array_replace — Replaces elements from passed arrays into the first array
- array_reverse — 원소를 역순으로 가지는 배열을 반환
- array_search — 주어진 값으로 배열을 검색하여 성공시 해당하는 키를 반환
- array_shift — 배열의 맨 앞에 있는 원소를 시프트
- array_slice — 배열의 일부를 추출
- array_splice — 배열의 일부를 삭제하고, 그 위치를 다른 내용으로 대체
- array_sum — 배열 값들의 합을 계산
- array_udiff_assoc — Computes the difference of arrays with additional index check, compares data by a callback function
- array_udiff_uassoc — Computes the difference of arrays with additional index check, compares data and indexes by a callback function
- array_udiff — 데이터 비교 콜백함수를 사용하여 배열간의 차이를 계산
- array_uintersect_assoc — Computes the intersection of arrays with additional index check, compares data by a callback function
- array_uintersect_uassoc — Computes the intersection of arrays with additional index check, compares data and indexes by separate callback functions
- array_uintersect — Computes the intersection of arrays, compares data by a callback function
- array_unique — 배열에서 중복된 값을 제거
- array_unshift — 배열의 맨 앞에 하나 이상의 원소를 첨가
- array_values — 배열의 모든 값을 반환
- array_walk_recursive — Apply a user function recursively to every member of an array
- array_walk — 배열의 각 원소에 대해서 특정 함수를 적용
- array — 배열 생성
- arsort — 배열을 내림차순 정렬하고 인덱스의 상관관계를 유지
- asort — 배열을 정렬하고 인덱스 상관 관계를 유지
- compact — 변수와 그 값을 가지는 배열 생성
- count — 배열의 모든 원소나, 객체의 프로퍼티 수를 셉니다
- current — 배열의 현재 원소를 반환
- each — 배열에서 현재 키와 값 쌍을 반환하고 배열 커서를 전진
- end — 배열 내부 포인터가 마지막 원소를 가리키게 설정
- extract — 배열에서 현재 심볼 테이블로 변수를 입력
- in_array — 값이 배열 안에 존재하는지 확인
- key_exists — 별칭: array_key_exists
- key — 배열에서 키를 가져옵니다
- krsort — 키에 의한 배열 역순 정렬
- ksort — 키에 의한 배열 정렬
- list — 배열처럼 변수에 할당
- natcasesort — "자연순" 알고리즘으로 대소문자를 구분하지 않고 배열 정렬
- natsort — "자연순" 알고리즘으로 배열 정렬
- next — 배열의 내부 배열 포인터를 전진
- pos — 별칭: current
- prev — 내부 배열 포인터를 후진
- range — 원소의 범위를 가지는 배열 생성
- reset — 배열의 내부 포인터를 첫 원소로 설정
- rsort — 역순으로 배열 정렬

- shuffle — 배열을 섞습니다
- sizeof — 별칭: count
- sort — 배열 정렬
- uasort — 사용자 정의 비교 함수로 배열을 정렬하고 인덱스 연관성을 유지
- uksort — 사용자 정의 비교 함수를 사용하여 키에 의한 배열 정렬
- usort — 사용자 정의 비교 함수를 사용하여 값에 의한 배열 정렬

add a note

User Contributed Notes 16 notes

up
down
81
applegrew at rediffmail dot com ¶
9 years ago

```
For newbies like me.

Creating new arrays:-
//Creates a blank array.
$theVariable = array();

//Creates an array with elements.
$theVariable = array("A", "B", "C");

//Creating Associaive array.
$theVariable = array(1 => "http//google.com", 2=> "http://yahoo.com");

//Creating Associaive array with named keys
$theVariable = array("google" => "http//google.com", "yahoo"=> "http://yahoo.com");

Note:
New value can be added to the array as shown below.
$theVariable[] = "D";
$theVariable[] = "E";
```

up
down
7
Tyler Bannister ¶
8 years ago

```
To delete an individual array element use the unset function

For example:

<?PHP
    $arr = array( "A", "B", "C" );
    unset( $arr[1] );
    // now $arr = array( "A", "C" );
?>

Unlink is for deleting files.
```

up
down
4
dragos dot rusu at NOSPAM dot bytex dot ro ¶

7 years ago

If an array item is declared with key as NULL, array key will automatically be converted to empty string '', as follows:

```
<?php
$a = array(
    NULL => 'zero',
    1    => 'one',
    2    => 'two');

// This will show empty string for key associated with "zero" value
var_dump(array_keys($a));

// Array elements are shown
reset($a);
while( key($a) !== NULL )
{
  echo key($a) . ": ".current($a) . "<br>";// PHP_EOL
  next($a);
}

// Array elements are not shown
reset($a);
while( key($a) != NULL ) // '' == null   => no iteration will be executed
{
  echo key($a) . ": ".current($a) . "<br>";// PHP_EOL
  next($a);
}
```

up
down
0

web at houhejie dot cn ¶

1 day ago

```
string2array($str):

$arr=json_decode('["fileno",["uid","uname"],"topingid",["touid",[1,2,[3,4]],"touname"]]');
print_r($arr);

output:
Array ( [0] => fileno [1] => Array ( [0] => uid [1] => uname ) [2] => topingid [3] =>
Array ( [0] => touid [1] => Array ( [0] => 1 [1] => 2 [2] => Array ( [0] => 3 [1] => 4 ) )
[2] => touname ) )

when I hope a function string2array($str), "spam2"  suggest this. and It works well~~~hope
this helps us,  and add to the Array function list
```

up
down
-2

macnimble at gmail dot com ¶

8 years ago

```
Converting a linear array (like a mysql record set) into a tree, or multi-dimensional
array can be a real bugbear. Capitalizing on references in PHP, we can 'stack' an array in
one pass, using one loop, like this:

<?php
```

```
# array_stack()
# Original idea from:
# http://www.ideashower.com/our_solutions/
#   create-a-parent-child-array-structure-in-one-pass/
function array_stack (&$a, $p = '@parent', $c = '@children')
{
  $l = $t = array();
  foreach ($a AS $key => $val):
    if (!$val[$p]) $t[$key] =& $l[$key];
    else $l[$val[$p]][$c][$key] =& $l[$key];
    $l[$key] = (array)$l[$key] + $val;
  endforeach;
  return $a = array('tree' => $t, 'leaf' => $l);
}

# Example:
$node = array();
$node[1] = array('@parent' => 0, 'title' => 'I am node 1.');
#     ^-----------------------v Link @parent value to key.
$node[2] = array('@parent' => 1, 'title' => 'I am node 2.');
$node[3] = array('@parent' => 2, 'title' => 'I am node 3.');
$node[4] = array('@parent' => 1, 'title' => 'I am node 4.');
$node[5] = array('@parent' => 4, 'title' => 'I am node 5.');

array_stack($node);

$node['leaf'][1]['title'] = 'I am node one.';
$node['leaf'][2]['title'] = 'I am node two.';
$node['leaf'][3]['title'] = 'I am node three.';
$node['leaf'][4]['title'] = 'I am node four.';
$node['leaf'][5]['title'] = 'I am node five.';

echo '<pre>',print_r($node['tree'],TRUE),'</pre>';
?>
```

Note that there's no parameter checking on the array value, but this is only to keep the function size small. One could easily a quick check in there to make sure the $a parameter was in fact an array.

Hope you find it useful. Huge thanks to Nate Weiner of IdeaShower.com for providing the original function I built on.

up
down
-4
contact at greyphoenix dot biz ¶
9 years ago

```
<?php
//Creating a multidimensional array

$theVariable = array("Search Engines" =>
array (
    0=> "http//google.com",
    1=> "http//yahoo.com",
    2=> "http//msn.com/"),
```

```
"Social Networking Sites" =>
array (
    0 => "http//www.facebook.com",
    1 => "http//www.myspace.com",
    2 => "http//vkontakte.ru",)
);

echo "The first array value is " . $theVariable['Search Engines'][0];
?>

-- Output--
The first array value is http://google.com
```

up
down
-5
Anonymous ¶
9 years ago

```
@jorge at andrade dot cl
This variant is faster:
<?php
function array_avg($array,$precision=2){
    if(!is_array($array))
        return 'ERROR in function array_avg(): this is a not array';

    foreach($array as $value)
        if(!is_numeric($value))
            return 'ERROR in function array_avg(): the array contains one or more non-
numeric values';

    $cuantos=count($array);
  return round(array_sum($array)/$cuantos,$precision);
}
?>
```

up
down
-6
info at curtinsNOSPAMcreations dot com ¶
7 years ago

```
Another way to create a multidimensional array that looks a lot cleaner is to use
json_decode.  (Note that this probably adds a touch of overhead, but it sure does look
nicer.)  You can of course add as many levels and as much formatting as you'd like to the
string you then decode.  Don't forget that json requires " around values, not '!!  (So,
you can't enclose the json string with " and use ' inside the string.)

As an example:

<?php
    $myarray['blah'] = json_decode('[
        {"label":"foo","name":"baz"},
        {"label":"boop","name":"beep"}
    ]',true);

print_r($myarray)
?>
returns:
```

```
Array
(
    [blah] => Array
        (
            [0] => Array
                (
                    [label] => foo
                    [name] => baz
                )

            [1] => Array
                (
                    [label] => boop
                    [name] => beep
                )
        )
)
```

up
down
-4

andyd273 at gmail dot com ¶

8 years ago

A small correction to Endel Dreyer's PHP array to javascript array function. I just changed it to show keys correctly:

```
            function array2js($array,$show_keys)
            {
                $dimensoes = array();
                $valores = array();

                $total = count ($array)-1;
                $i=0;
                foreach($array as $key=>$value){
                    if (is_array($value)) {
                        $dimensoes[$i] = array2js($value,$show_keys);
                        if ($show_keys) $dimensoes[$i] = '"'.$key.'":'.$dimensoes[$i];
                    } else {
                        $dimensoes[$i] = '"'.addslashes($value).'"';
                        if ($show_keys) $dimensoes[$i] = '"'.$key.'":'.$dimensoes[$i];
                    }
                    if ($i==0) $dimensoes[$i] = '{'.$dimensoes[$i];
                    if ($i==$total) $dimensoes[$i].= '}';
                    $i++;
                }
                return implode(',',$dimensoes);
            }
```

up
down
-3

webmaster at infoproducts dot x10hosting dot com ¶

9 years ago

New value can also be added to the array as shown below.

```
$theVariable["google"] = "http//google.com";
```

```
or
$theVariable["1"] = "http//google.com";
```

up
down
-7
spereversev at envionsoftware dot com ¶
5 years ago

```
<?php
function array_mask(array $array, array $keys) {
    return array_intersect_key( $array, array_fill_keys( $keys, 0 ) );
}
?>
```

Might be helpful to take a part of associative array containing given keys, for example, from a $_REQUEST array

```
array_mask($_REQUEST, array('name', 'email'));
```

up
down
-6
sunear at gmail dot com ¶
8 years ago

Made this function to delete elements in an array;

```
<?php

function array_del_elm($input_array, $del_indexes) {
    if (is_array($del_indexes)) {
        $indexes = $del_indexes;
    } elseif(is_string($del_indexes)) {
        $indexes = explode($del_indexes, " ");
    } elseif(is_numeric($del_indexes)) {
        $indexes[0] = (integer)$del_indexes;
    } else return;
    $del_indexes = null;

    $cur_index = 0;
    if (sort($indexes)) for($i=0; $i<count($input_array); $i++) {
        if ($i == $indexes[$cur_index]) {
            $cur_index++;
            if ($cur_index == count($indexes)) return $output_array;
            continue;
        }
        $output_array[] = $input_array[$i];
    }
    return $output_array;
}

?>
```

but then i saw the methods of doing the same by Tyler Bannister & Paul, could see that theirs were faster, but had floors regarding deleting multiple elements thus support of several ways of giving parameters. I combined the two methods to this to this:

```
<?php
```

```
function array_del_elm($target_array, $del_indexes) {
    if (is_array($del_indexes)) {
        $indexes = $del_indexes;
    } elseif(is_string($del_indexes)) {
        $indexes = explode($del_indexes, " ");
    } elseif(is_numeric($del_indexes)) {
        $indexes[0] = (integer)$del_indexes;
    } else return;
    unset($del_indexes);

    for($i=0; $i<count($indexes); $i++) {
        unset($target_array[$indexes[$i]]);
    }
    return $target_array;
}

?>

Fast, compliant and functional ;)
```

up
down
-9

Jack A ¶
9 years ago

```
Note that arrays are not allowed in class constants and trying to do so will throw a fatal
error.
```

up
down
-8

gratcypalma at gmail dot com ¶
4 years ago

```
<?php
function foo() {
    return array('name' => 'palma', 'old' => 23, 'language' => 'PHP');
}
/* 1. PHP < 5.4.0 */
$a = foo();
var_dump($a['name']);

/* 2. Works ini PHP >= 5.4.0 */

var_dump(foo()['name']);

/*
When i run second method on PHP 5.3.8 i will be show error message "PHP Fatal error:
Can't use method return value in write context"

http://www.php.net/manual/en/migration54.new-features.php
*/
```

up
down
-11

thomasdecaux at ebuildy dot com ¶
8 years ago

```
To browse a simple array:

<?php

foreach ($myArray AS $myItem)
{

}

?>

To browse an associative array:

<?php

foreach ($myArray AS $key=>$value)
{

}

?>
```

http://www.ebuildy.com
up
down
-23
John Marc ¶
7 years ago

```
Be careful when adding elements to a numeric array.
I wanted to store some info about some items from a database and decided to use the record
id as a key.

<?php
$key=3000000000;
$DATA[$key]=true;
?>

This will create an array of 30 million elements and chances are, you will use up all
memory with these 2 lines

<?php
$key=3000000000;
$DATA["$key"]=true;
?>

This on the other hand will force the array to be an associative array and will only
create the one element
```

add a note

- Variable and Type Related Extensions
  - 배열
  - 클래스/객체
  - Classkit
  - Ctype
  - Data Structures

    - Filter
    - Function Handling
    - Quickhash
    - Reflection
    - Variable handling

- My PHP.net
- Contact
- Other PHP.net sites
- Mirror sites
- Privacy policy