Focus search box

array_splice »
« array_shift

- Manual de PHP
- Referencia de funciones
- Extensiones relacionadas con variable y tipo
- Arrays
- Funciones de Arrays

Change language: Spanish

Submit a Pull Request Report a Bug

array_slice

(PHP 4, PHP 5, PHP 7, PHP 8)

array_slice — Extraer una parte de un array

Descripción ¶

```
array_slice(
    array $array,
    int $offset,
    int $length = null,
    bool $preserve_keys = false
): array
```

array_slice() devuelve la secuencia de elementos del array `array` tal y como se especifica en los parámetros `offset` y `length`.

Parámetros ¶

`array`

El array de entrada.

`offset`

Si el índice dado por `offset` no es negativo, la secuencia empezará en esa posición del `array`. Si el `offset` es negativo, la secuencia empezará en esa posición empezando por el final del `array`.

`length`

Si la longitud dada por `length` es positiva, la secuencia tendrá hasta tantos elementos como indique el valor. Si el array es más corto que `length`, solamente estarán presentes los elementos disponibles del array. Si se proporciona `length` y es negativo, la secuencia finalizará en tantos elementos empezando por el final del array. Si se omite, entonces la secuencia contendrá todo el contenido desde `offset` hasta el final del `array`.

`preserve_keys`

Observe que **array_slice()** reordenará y reinicializará los índices numéricos del array de forma predeterminada. Se puede cambiar esta comportamiento estableciendo el parámetro `preserve_keys` a **`true`**.

Valores devueltos ¶

Devuelve la parte del array. Si el índice es mayor que el tamaño del array, devuelve un array vacío.

Historial de cambios ¶

Versión	Descripción
5.2.4	El valor predeterminado del parámetro `length` se cambió a `NULL`. Una `length` `NULL` ahora indica a la función que use la longitud de `array`. Antes de esta versión, una `length` `NULL` se tomaba como de longitud cero (no se devolvía nada).
5.0.2	Se añadió el parámetro opcional `preserve_keys`.

Ejemplos ¶

Ejemplo #1 Ejemplos de array_slice()

```
<?php
$entrada = array("a", "b", "c", "d", "e");

$salida = array_slice($entrada, 2);      // devuelve "c", "d", y "e"
$salida = array_slice($entrada, -2, 1);  // devuelve "d"
$salida = array_slice($entrada, 0, 3);   // devuelve "a", "b", y "c"

// observe las diferencias en las claves de los arrays
print_r(array_slice($entrada, 2, -1));
print_r(array_slice($entrada, 2, -1, true));
?>
```

El resultado del ejemplo sería:

```
Array
(
    [0] => c
    [1] => d
)
Array
(
    [2] => c
    [3] => d
)
```

Ver también ¶

- array_splice() - Elimina una porción del array y la reemplaza con otra cosa
- unset() - Destruye una o más variables especificadas
- array_chunk() - Divide un array en fragmentos

+ add a note

User Contributed Notes 19 notes

up
down
43
taylorbarstow at the google mail service ¶
16 years ago

Array slice function that works with associative arrays (keys):

```
function array_slice_assoc($array,$keys) {
    return array_intersect_key($array,array_flip($keys));
}
```

up
down
2
nathan dot fiscaletti at gmail dot com ¶
4 years ago

If you want an associative version of this you can do the following:

```
function array_slice_assoc($array,$keys) {
    return array_intersect_key($array,array_flip($keys));
}
```

However, if you want an inverse associative version of this, just use array_diff_key instead of array_intersect_key.

```
function array_slice_assoc_inverse($array,$keys) {
    return array_diff_key($array,array_flip($keys));
}
```

Example:

```
$arr = [
    'name' => 'Nathan',
    'age' => 20,
    'height' => 6
];

array_slice_assoc($arr, ['name','age']);
```

will return

```
Array (
     'name' = 'Nathan',
     'age' = 20
)
```

Where as

```
array_slice_assoc_inverse($arr, ['name']);
```

will return

```
Array (
    'age' = 20,
    'height' = 6
)
```

up
down
13
Ray.Paseur often uses Gmail ¶
9 years ago

```
<?php
// CHOP $num ELEMENTS OFF THE FRONT OF AN ARRAY
// RETURN THE CHOP, SHORTENING THE SUBJECT ARRAY
```

```
function array_chop(&$arr, $num)
{
    $ret = array_slice($arr, 0, $num);
    $arr = array_slice($arr, $num);
    return $ret;
}
```

up
down
11
developer at i-space dot org ¶
20 years ago
remember that array_slice returns an array with the current element. you must use array_slice($array, $index+1) if you want to get the next elements.
up
down
12
worldclimb at 99gmail99 dot com ¶
14 years ago
array_slice can be used to remove elements from an array but it's pretty simple to use a custom function.

One day array_remove() might become part of PHP and will likely be a reserved function name, hence the unobvious choice for this function's names.

```
<?
function arem($array,$value){
    $holding=array();
    foreach($array as $k => $v){
        if($value!=$v){
            $holding[$k]=$v;
        }
    }
    return $holding;
}

function akrem($array,$key){
    $holding=array();
    foreach($array as $k => $v){
        if($key!=$k){
            $holding[$k]=$v;
        }
    }
    return $holding;
}

$lunch = array('sandwich' => 'cheese', 'cookie'=>'oatmeal','drink' => 'tea','fruit' => 'apple');
echo '<pre>';
print_r($lunch);
$lunch=arem($lunch,'apple');
print_r($lunch);
$lunch=akrem($lunch,'sandwich');
print_r($lunch);
echo '</pre>';
?>
```

(remove 9's in email)
up
down

6

s0i0m at dreamevilconcepts dot com ¶

14 years ago

Using the varname function referenced from the array_search page, submitted by dcez at land dot ru. I created a multi-dimensional array splice function. It's usage is like so:

```
$array['admin'] = array('blah1', 'blah2');
$array['voice'] = array('blah3', 'blah4');
array_cut('blah4', $array);
```

...Would strip blah4 from the array, no matter where the position of it was in the array ^^ Returning this...

```
Array ( [admin] => Array ( [0] => blah1 [1] => blah2 ) [voice] => Array ( [0] => blah3 ) )
```

Here is the code...

```
<?php

  function varname ($var)
  {
    // varname function by dcez at land dot ru
    return (isset($var)) ? array_search($var, $GLOBALS) : false;
  }

  function array_cut($needle, $haystack)
  {
    foreach ($haystack as $k => $v)
    {
      for ($i=0; $i<count($v); $i++)
        if ($v[$i] === $needle)
        {
          return array_splice($GLOBALS[varname($haystack)][$k], $i, 1);
          break; break;
        }
    }

?>
```

Check out dreamevilconcept's forum for more innovative creations!

up
down

1

ted.devito at 9gmail9 dot 99com ¶

14 years ago

based on worldclimb's arem(), here is a recursive array value removal tool that can work with multidimensional arrays.

```
function remove_from_array($array,$value){
    $clear = true;
    $holding=array();

    foreach($array as $k => $v){
        if (is_array($v)) {
            $holding [$k] = remove_from_array ($v, $value);
            }
        elseif ($value == $v) {
            $clear = false;
```

```
            }
        elseif($value != $v){
            $holding[$k]=$v; // removes an item by combing through the array in order and saving
the good stuff
        }
    }
    if ($clear) return $holding; // only pass back the holding array if we didn't find the value
}
```

up
down
0

kansey ¶

7 years ago

To save the sort order of a numeric index in the array. Version php =>5.5.26

```
/*
Example
*/

$arr = array( "1" =>2, "2" =>3 , "3" =>5 );

print_r(array_slice($arr,1,null,true));

/*
Result

Array
(
[2] => 3
[3] => 5
)
*/
```

up
down
-1

xananax at yelostudio dot com ¶

11 years ago

```
<?php
/**
* Reorders an array by keys according to a list of values.
* @param array $array the array to reorder. Passed by reference
* @param array $list the list to reorder by
* @param boolean $keepRest if set to FALSE, anything not in the $list array will be removed.
* @param boolean $prepend if set to TRUE, will prepend the remaining values instead of appending
them
* @author xananax AT yelostudio DOT com
*/
function array_reorder(array &$array,array $list,$keepRest=TRUE,$prepend=FALSE,$preserveKeys=TRUE)
{
    $temp = array();
    foreach($list as $i){
        if(isset($array[$i])){
            $tempValue = array_slice(
                $array,
                array_search($i,array_keys($array)),
                1,
                $preserveKeys
            );
            $temp[$i] = array_shift($tempValue);
```

```
            unset($array[$i]);
        }
    }
    $array = $keepRest ?
        ($prepend?
            $array+$temp
            :$temp+$array
        )
        : $temp;
}

/** exemple ** /
$a = array(
    'a'    =>    'a',
    'b'    =>    'b',
    'c'    =>    'c',
    'd'    =>    'd',
    'e'    =>    'e'
);
$order = array('c','b','a');

array_reorder($a,$order,TRUE);
echo '<pre>';
print_r($a);
echo '</pre>';
/** exemple end **/
?>
```

up
down
-2

andreasblixt (at) msn (dot) com ¶

17 years ago

```
<?php
    // Combines two arrays by inserting one into the other at a given position then returns the
result
    function array_insert($src, $dest, $pos) {
        if (!is_array($src) || !is_array($dest) || $pos <= 0) return FALSE;
        return array_merge(array_slice($dest, 0, $pos), $src, array_slice($dest, $pos));
    }
?>
```

up
down
-4

aexchecker at yahoo dot com ¶

15 years ago

```
<?php
/**
* @desc
* Combines two arrays by inserting one into the other at a given position then
* returns the result.
*
* @since   2007/10/04
* @version v0.7 2007/10/04 18:47:52
* @author  AexChecker <AexChecker@yahoo.com>
* @param   array $source
* @param   array $destination
* @param   int [optional] $offset
* @param   int [optional] $length
```

```
 * @return  array
 */
function array_insert($source, $destination, $offset = NULL, $length = NULL) {
    if (!is_array($source) || empty($source)) {
        if (is_array($destination) && !empty($destination)) {
            return $destination;
        }
        return array();
    }
    if (is_null($offset)) {
        return array_merge($destination, $source);
    }
    $offset = var2int($offset);
    if (is_null($length)) {
        if ($offset === 0) {
            return array_merge($source, array_slice($destination, 1));
        }
        if ($offset === -1) {
            return array_merge(array_slice($destination, 0, -1), $source);
        }
        return array_merge(
            array_slice($destination, 0, $offset),
            $source,
            array_slice($destination, ++$offset)
        );
    }
    if ($offset === 0) {
        return array_merge($source, array_slice($destination, $length));
    }
    $destination_count = count($destination);
    $length = var2int($length);
    if ($offset > 0) {
        if ($destination_count - $offset < 1) {
            return array_merge($destination, $source);
        }
    } else{
        if (($t = $destination_count + $offset) < 1) {
            return array_merge($source, $destination);
        }
        $offset = $t;
    }
    if ($length > 0) {
        $length+= $offset;
    } elseif ($length < 0 && !($length * -1 < $destination_count)) {
        return $source;
    } else {
        $length = $offset;
    }
    return array_merge(
        array_slice($destination, 0, $offset),
        $source,
        array_slice($destination, $length)
    );
}
?>
```

up
down
-3

delew ¶
11 years ago
just a little tip.
to preserve keys without providing length: use NULL

```
array_slice($array, $my_offset, NULL, true);
```

up
down
-4

aflavio at gmail dot com ¶
15 years ago

```
/**
    * Remove a value from a array
    * @param string $val
    * @param array $arr
    * @return array $array_remval
    */
    function array_remval($val, &$arr)
    {
          $array_remval = $arr;
          for($x=0;$x<count($array_remval);$x++)
          {
              $i=array_search($val,$array_remval);
              if (is_numeric($i)) {
                  $array_temp  = array_slice($array_remval, 0, $i );
                $array_temp2 = array_slice($array_remval, $i+1, count($array_remval)-1 );
                $array_remval = array_merge($array_temp, $array_temp2);
              }
          }
          return $array_remval;
    }

$stack=Array('apple','banana','pear','apple', 'cherry', 'apple');
array_remval("apple", $stack);

//output: Array('banana','pear', 'cherry')
```

up
down
-2

Anonymous ¶
16 years ago
If you specify the fourth argument (to not reassign the keys), then there appears to be no way to get the function to return all values to the end of the array. Assigning -0 or NULL or just putting two commas in a row won't return any results.

up
down
-5

jamon at clearsightdesign dot com ¶
13 years ago
I was trying to find a good way to find the previous several and next several results from an array created in a MySQL query. I found that most MySQL solutions to this problem were complex. Here is a simple function that returns the previous and next rows from the array.

```
<?php

/*
** function array_surround by Jamon Holmgren of ClearSight Design
** Version 1.0 - 4/10/2009
```

```
** Please direct comments and questions to my first name at symbol clearsightdesign.com
**
** Returns an array with only the $before and $after number of results
** This is set to work best with MySQL data results
** Use this to find the rows immediately before and after a particular row, as many as you want
**
** Example usage:
**   $mysql_ar is an array of results from a MySQL query and the current id is $cur_id
**   We want to get the row before this one and five rows afterward
**
** $near_rows = array_surround($mysql_ar, "id", $cur_id, 1, 5)
**
**   Previous row is now $near_rows[-1]
**   Current row is now $near_rows[0]
**   Next row is $near_rows[1] ... etc
**   If there is no previous row, $near_rows[-1] will not be set...test for it with
is_array($near_rows[-1])
**
*/
function array_surround($src_array, $field, $value, $before = 1, $after = 1) {
    if(is_array($src_array)) {
        // reset all the keys to 0 through whatever in case they aren't sequential
        $new_array = array_values($src_array);
        // now loop through and find the key in array that matches the criteria in $field and
$value
        foreach($new_array as $k => $s) {
            if($s[$field] == $value) {
                // Found the one we wanted
                $ck = $k; // put the key in the $ck (current key)
                break;
            }
        }
        if(isset($ck)) { // Found it!
            $result_start = $ck - $before; // Set the start key
            $result_length = $before + 1 + $after; // Set the number of keys to return
            if($result_start < 0) { // Oops, start key is before first result
                $result_length = $result_length + $result_start; // Reduce the number of keys to
return
                $result_start = 0; // Set the start key to the first result
            }
            $result_temp = array_slice($new_array, $result_start, $result_length); // Slice out
the results we want
            // Now we have an array, but we want array[-$before] to array[$after] not 0 to
whatever.
            foreach($result_temp as $rk => $rt) { // set all the keys to -$before to +$after
                $result[$result_start - $ck + $rk] = $rt;
            }
            return $result;
        } else { // didn't find it!
            return false;
        }
    } else { // They didn't send an array
        return false;
    }
}

?>
```

```
I hope you find this useful! I welcome constructive criticism or comments or of course praise ;) -
- just e-mail me.

- Jamon Holmgren
```

up
down
-6
Nathan - thefiscster510 at gmail dot com ¶
11 years ago

```
If you want to remove a specified entry from an array i made this mwethod...

<?php
$array = array("Entry1","entry2","entry3");

$int = 3; //Number of entries in the array
$int2 = 0; //Starter array spot... it will begine its search at 0.
$del_num = 1; //Represents the second entry in the array... which is the one we will happen to
remove this time... i.e. 0 = first entry, 1 = second entry, 2 = third..........

$newarray = array(); //Empty array that will be the new array minus the specified entry...
print_r($array) . "<br>";    //print original array contents
print_r($newarray). "<br>"; //print the new empty array

do
{
$user = $array[$int2];
$key = array_search($user, $array);
if ($key == $del_num)
{

}
else
{
$newarray[] = $array[$int2];
}

$int2 = $int2 + 1;
} while ($int2 < $int);

print_r($newarray). "<br>"; //print the new array

?>
```

up
down
-3
bishop ¶
17 years ago

```
Sometimes you need to pick certain non-integer and/or non-sequential keys out of an array.
Consider using the array_pick() implementation below to pull specific keys, in a specific order,
out of a source array:

<?php

$a = array ('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4);
$b = array_pick($a, array ('d', 'b'));

// now:
// $a = array ('a' => 1, 'c' => '3');
```

```
// $b = array ('d' => 4, 'b' => '2');

function &array_pick(&$array, $keys)
{
    if (! is_array($array)) {
        trigger_error('First parameter must be an array', E_USER_ERROR);
        return false;
    }

    if (! (is_array($keys) || is_scalar($keys))) {
        trigger_error('Second parameter must be an array of keys or a scalar key', E_USER_ERROR);
        return false;
    }

    if (is_array($keys)) {
        // nothing to do
    } else if (is_scalar($keys)) {
        $keys = array ($keys);
    }

    $resultArray = array ();
    foreach ($keys as $key) {
        if (is_scalar($key)) {
            if (array_key_exists($key, $array)) {
                $resultArray[$key] = $array[$key];
                unset($array[$key]);
            }
        } else {
            trigger_error('Supplied key is not scalar', E_USER_ERROR);
            return false;
        }
    }

    return $resultArray;
}

?>
```

up
down
-4
Mr. P ¶
14 years ago

Note that offset is not the same thing as key. Offset always starts at 0, while keys might be any number.

So this:

```
<?php print_r(array_slice(array(0 => 0, 5 => 5, 13 => 13),1)); ?>
```

will result in this:

```
Array
(
    [0] => 5
    [1] => 13
)
```

up
down
-5

SomeGuy ¶
6 years ago

```
Thank to taylorbarstow here the function with the unset feature.
<?php
function array_slice_assoc(&$array,$keys,$unset = true) {
    $return = array_intersect_key($array, array_flip($keys));
    if ($unset) {
        foreach ($keys as $value) {
            unset($array[$value]);
        }
    }
    return $return;
}
?>
```

+ add a note