[**PHP Array 函数** 2](#_Toc278904028)

[array() 6](#_Toc278904029)

[array\_change\_key\_case() 函数 7](#_Toc278904030)

[array\_chunk() 函数 8](#_Toc278904031)

[array\_combine() 函数 10](#_Toc278904032)

[array\_count\_values() 函数 10](#_Toc278904033)

[array\_diff() 函数 11](#_Toc278904034)

[array\_diff\_assoc() 函数 12](#_Toc278904035)

[array\_diff\_key() 函数 13](#_Toc278904036)

[array\_diff\_uassoc() 函数 14](#_Toc278904037)

[array\_diff\_ukey() 函数 16](#_Toc278904038)

[array\_fill() 函数 18](#_Toc278904039)

[array\_filter() 函数 19](#_Toc278904040)

[array\_flip() 函数 20](#_Toc278904041)

[array\_intersect() 函数 21](#_Toc278904042)

[array\_intersect\_assoc() 函数 21](#_Toc278904043)

[array\_intersect\_key() 函数 23](#_Toc278904044)

[array\_intersect\_uassoc() 函数 24](#_Toc278904045)

[array\_intersect\_ukey() 函数 26](#_Toc278904046)

[array\_key\_exists() 函数 28](#_Toc278904047)

[array\_keys() 函数 29](#_Toc278904048)

[array\_map() 函数 31](#_Toc278904049)

[array\_merge() 函数 33](#_Toc278904050)

[array\_merge\_recursive() 函数 34](#_Toc278904051)

[array\_multisort() 函数 35](#_Toc278904052)

[array\_pad() 函数 38](#_Toc278904053)

[array\_pop() 函数 39](#_Toc278904054)

[array\_product() 函数 40](#_Toc278904055)

[array\_push() 函数 40](#_Toc278904056)

[array\_rand() 函数 42](#_Toc278904057)

[array\_reduce() 函数 43](#_Toc278904058)

[array\_reverse() 函数 44](#_Toc278904059)

[array\_search() 函数 45](#_Toc278904060)

[array\_shift() 函数 46](#_Toc278904061)

[array\_slice() 函数 48](#_Toc278904062)

[array\_splice() 函数 49](#_Toc278904063)

[array\_sum() 函数 51](#_Toc278904064)

[array\_udiff() 函数 52](#_Toc278904065)

[array\_udiff\_assoc() 函数 53](#_Toc278904066)

[array\_udiff\_uassoc() 函数 55](#_Toc278904067)

[array\_uintersect() 函数 56](#_Toc278904068)

[array\_uintersect\_assoc() 函数 58](#_Toc278904069)

[array\_uintersect\_uassoc() 函数 59](#_Toc278904070)

[array\_unique() 函数 61](#_Toc278904071)

[array\_unshift() 函数 62](#_Toc278904072)

[array\_values() 函数 63](#_Toc278904073)

[array\_walk() 函数 64](#_Toc278904074)

[array\_walk\_recursive() 函数 66](#_Toc278904075)

[arsort() 函数 68](#_Toc278904076)

[asort() 函数 69](#_Toc278904077)

[compact() 函数 70](#_Toc278904078)

[count() 函数 72](#_Toc278904079)

[PHP Array 函数current() 函数 73](#_Toc278904080)

[each() 函数 74](#_Toc278904081)

[each() 函数 75](#_Toc278904082)

[extract() 函数 76](#_Toc278904083)

[in\_array() 函数 78](#_Toc278904084)

[key() 函数 80](#_Toc278904085)

[krsort() 函数 81](#_Toc278904086)

[ksort() 函数 82](#_Toc278904087)

[list() 函数 83](#_Toc278904088)

[natcasesort() 函数 84](#_Toc278904089)

[natsort() 函数 86](#_Toc278904090)

[next() 函数 87](#_Toc278904091)

[pos() 函数 88](#_Toc278904092)

[prev() 函数 89](#_Toc278904093)

[PHP Array 函数range() 函数 90](#_Toc278904094)

[reset() 函数 92](#_Toc278904095)

[rsort() 函数 93](#_Toc278904096)

[shuffle() 函数 94](#_Toc278904097)

[sizeof() 函数 95](#_Toc278904098)

[sort() 函数 96](#_Toc278904099)

[uasort() 函数 97](#_Toc278904100)

[uksort() 函数 99](#_Toc278904101)

[usort() 函数 100](#_Toc278904102)

**PHP Array 函数**

* [Previous Page](http://www.w3school.com.cn/php/php_ajax_poll.asp)
* [Next Page](http://www.w3school.com.cn/php/php_ref_calendar.asp)

**PHP Array 简介**

array 函数允许您对数组进行操作。

PHP 支持单维和多维的数组。同时提供了用数据库查询结果来构造数组的函数。

**安装**

array 函数是 PHP 核心的组成部分。无需安装即可使用这些函数。

**PHP Array 函数**

PHP：指示支持该函数的最早的 PHP 版本。

|  |  |  |
| --- | --- | --- |
| **函数** | **描述** | **PHP** |
| [array()](http://www.w3school.com.cn/php/func_array.asp) | 创建数组。 | 3 |
| [array\_change\_key\_case()](http://www.w3school.com.cn/php/func_array_change_key_case.asp) | 返回其键均为大写或小写的数组。 | 4 |
| [array\_chunk()](http://www.w3school.com.cn/php/func_array_chunk.asp) | 把一个数组分割为新的数组块。 | 4 |
| [array\_combine()](http://www.w3school.com.cn/php/func_array_combine.asp) | 通过合并两个数组来创建一个新数组。 | 5 |
| [array\_count\_values()](http://www.w3school.com.cn/php/func_array_count_values.asp) | 用于统计数组中所有值出现的次数。 | 4 |
| [array\_diff()](http://www.w3school.com.cn/php/func_array_diff.asp) | 返回两个数组的差集数组。 | 4 |
| [array\_diff\_assoc()](http://www.w3school.com.cn/php/func_array_diff_assoc.asp) | 比较键名和键值，并返回两个数组的差集数组。 | 4 |
| [array\_diff\_key()](http://www.w3school.com.cn/php/func_array_diff_key.asp) | 比较键名，并返回两个数组的差集数组。 | 5 |
| [array\_diff\_uassoc()](http://www.w3school.com.cn/php/func_array_diff_uassoc.asp) | 通过用户提供的回调函数做索引检查来计算数组的差集。 | 5 |
| [array\_diff\_ukey()](http://www.w3school.com.cn/php/func_array_diff_ukey.asp) | 用回调函数对键名比较计算数组的差集。 | 5 |
| [array\_fill()](http://www.w3school.com.cn/php/func_array_fill.asp) | 用给定的值填充数组。 | 4 |
| [array\_filter()](http://www.w3school.com.cn/php/func_array_filter.asp) | 用回调函数过滤数组中的元素。 | 4 |
| [array\_flip()](http://www.w3school.com.cn/php/func_array_flip.asp) | 交换数组中的键和值。 | 4 |
| [array\_intersect()](http://www.w3school.com.cn/php/func_array_intersect.asp) | 计算数组的交集。 | 4 |
| [array\_intersect\_assoc()](http://www.w3school.com.cn/php/func_array_intersect_assoc.asp) | 比较键名和键值，并返回两个数组的交集数组。 | 4 |
| [array\_intersect\_key()](http://www.w3school.com.cn/php/func_array_intersect_key.asp) | 使用键名比较计算数组的交集。 | 5 |
| [array\_intersect\_uassoc()](http://www.w3school.com.cn/php/func_array_intersect_uassoc.asp) | 带索引检查计算数组的交集，用回调函数比较索引。 | 5 |
| [array\_intersect\_ukey()](http://www.w3school.com.cn/php/func_array_intersect_ukey.asp) | 用回调函数比较键名来计算数组的交集。 | 5 |
| [array\_key\_exists()](http://www.w3school.com.cn/php/func_array_key_exists.asp) | 检查给定的键名或索引是否存在于数组中。 | 4 |
| [array\_keys()](http://www.w3school.com.cn/php/func_array_keys.asp) | 返回数组中所有的键名。 | 4 |
| [array\_map()](http://www.w3school.com.cn/php/func_array_map.asp) | 将回调函数作用到给定数组的单元上。 | 4 |
| [array\_merge()](http://www.w3school.com.cn/php/func_array_merge.asp) | 把一个或多个数组合并为一个数组。 | 4 |
| [array\_merge\_recursive()](http://www.w3school.com.cn/php/func_array_merge_recursive.asp) | 递归地合并一个或多个数组。 | 4 |
| [array\_multisort()](http://www.w3school.com.cn/php/func_array_multisort.asp) | 对多个数组或多维数组进行排序。 | 4 |
| [array\_pad()](http://www.w3school.com.cn/php/func_array_pad.asp) | 用值将数组填补到指定长度。 | 4 |
| [array\_pop()](http://www.w3school.com.cn/php/func_array_pop.asp) | 将数组最后一个单元弹出（出栈）。 | 4 |
| [array\_product()](http://www.w3school.com.cn/php/func_array_product.asp) | 计算数组中所有值的乘积。 | 5 |
| [array\_push()](http://www.w3school.com.cn/php/func_array_push.asp) | 将一个或多个单元（元素）压入数组的末尾（入栈）。 | 4 |
| [array\_rand()](http://www.w3school.com.cn/php/func_array_rand.asp) | 从数组中随机选出一个或多个元素，并返回。 | 4 |
| [array\_reduce()](http://www.w3school.com.cn/php/func_array_reduce.asp) | 用回调函数迭代地将数组简化为单一的值。 | 4 |
| [array\_reverse()](http://www.w3school.com.cn/php/func_array_reverse.asp) | 将原数组中的元素顺序翻转，创建新的数组并返回。 | 4 |
| [array\_search()](http://www.w3school.com.cn/php/func_array_search.asp) | 在数组中搜索给定的值，如果成功则返回相应的键名。 | 4 |
| [array\_shift()](http://www.w3school.com.cn/php/func_array_shift.asp) | 删除数组中的第一个元素，并返回被删除元素的值。 | 4 |
| [array\_slice()](http://www.w3school.com.cn/php/func_array_slice.asp) | 在数组中根据条件取出一段值，并返回。 | 4 |
| [array\_splice()](http://www.w3school.com.cn/php/func_array_splice.asp) | 把数组中的一部分去掉并用其它值取代。 | 4 |
| [array\_sum()](http://www.w3school.com.cn/php/func_array_sum.asp) | 计算数组中所有值的和。 | 4 |
| [array\_udiff()](http://www.w3school.com.cn/php/func_array_udiff.asp) | 用回调函数比较数据来计算数组的差集。 | 5 |
| [array\_udiff\_assoc()](http://www.w3school.com.cn/php/func_array_udiff_assoc.asp) | 带索引检查计算数组的差集，用回调函数比较数据。 | 5 |
| [array\_udiff\_uassoc()](http://www.w3school.com.cn/php/func_array_udiff_uassoc.asp) | 带索引检查计算数组的差集，用回调函数比较数据和索引。 | 5 |
| [array\_uintersect()](http://www.w3school.com.cn/php/func_array_uintersect.asp) | 计算数组的交集，用回调函数比较数据。 | 5 |
| [array\_uintersect\_assoc()](http://www.w3school.com.cn/php/func_array_uintersect_assoc.asp) | 带索引检查计算数组的交集，用回调函数比较数据。 | 5 |
| [array\_uintersect\_uassoc()](http://www.w3school.com.cn/php/func_array_uintersect_uassoc.asp) | 带索引检查计算数组的交集，用回调函数比较数据和索引。 | 5 |
| [array\_unique()](http://www.w3school.com.cn/php/func_array_unique.asp) | 删除数组中重复的值。 | 4 |
| [array\_unshift()](http://www.w3school.com.cn/php/func_array_unshift.asp) | 在数组开头插入一个或多个元素。 | 4 |
| [array\_values()](http://www.w3school.com.cn/php/func_array_values.asp) | 返回数组中所有的值。 | 4 |
| [array\_walk()](http://www.w3school.com.cn/php/func_array_walk.asp) | 对数组中的每个成员应用用户函数。 | 3 |
| [array\_walk\_recursive()](http://www.w3school.com.cn/php/func_array_walk_recursive.asp) | 对数组中的每个成员递归地应用用户函数。 | 5 |
| [arsort()](http://www.w3school.com.cn/php/func_array_arsort.asp) | 对数组进行逆向排序并保持索引关系。 | 3 |
| [asort()](http://www.w3school.com.cn/php/func_array_asort.asp) | 对数组进行排序并保持索引关系。 | 3 |
| [compact()](http://www.w3school.com.cn/php/func_array_compact.asp) | 建立一个数组，包括变量名和它们的值。 | 4 |
| [count()](http://www.w3school.com.cn/php/func_array_count.asp) | 计算数组中的元素数目或对象中的属性个数。 | 3 |
| [current()](http://www.w3school.com.cn/php/func_array_current.asp) | 返回数组中的当前元素。 | 3 |
| [each()](http://www.w3school.com.cn/php/func_array_each.asp) | 返回数组中当前的键／值对并将数组指针向前移动一步。 | 3 |
| [end()](http://www.w3school.com.cn/php/func_array_end.asp) | 将数组的内部指针指向最后一个元素。 | 3 |
| [extract()](http://www.w3school.com.cn/php/func_array_extract.asp) | 从数组中将变量导入到当前的符号表。 | 3 |
| [in\_array()](http://www.w3school.com.cn/php/func_array_in_array.asp) | 检查数组中是否存在指定的值。 | 4 |
| [key()](http://www.w3school.com.cn/php/func_array_key.asp) | 从关联数组中取得键名。 | 3 |
| [krsort()](http://www.w3school.com.cn/php/func_array_krsort.asp) | 对数组按照键名逆向排序。 | 3 |
| [ksort()](http://www.w3school.com.cn/php/func_array_ksort.asp) | 对数组按照键名排序。 | 3 |
| [list()](http://www.w3school.com.cn/php/func_array_list.asp) | 把数组中的值赋给一些变量。 | 3 |
| [natcasesort()](http://www.w3school.com.cn/php/func_array_natcasesort.asp) | 用“自然排序”算法对数组进行不区分大小写字母的排序。 | 4 |
| [natsort()](http://www.w3school.com.cn/php/func_array_natsort.asp) | 用“自然排序”算法对数组排序。 | 4 |
| [next()](http://www.w3school.com.cn/php/func_array_next.asp) | 将数组中的内部指针向前移动一位。 | 3 |
| [pos()](http://www.w3school.com.cn/php/func_array_pos.asp) | current() 的别名。 | 3 |
| [prev()](http://www.w3school.com.cn/php/func_array_prev.asp) | 将数组的内部指针倒回一位。 | 3 |
| [range()](http://www.w3school.com.cn/php/func_array_range.asp) | 建立一个包含指定范围的元素的数组。 | 3 |
| [reset()](http://www.w3school.com.cn/php/func_array_reset.asp) | 将数组的内部指针指向第一个元素。 | 3 |
| [rsort()](http://www.w3school.com.cn/php/func_array_rsort.asp) | 对数组逆向排序。 | 3 |
| [shuffle()](http://www.w3school.com.cn/php/func_array_shuffle.asp) | 把数组中的元素按随机顺序重新排列。 | 3 |
| [sizeof()](http://www.w3school.com.cn/php/func_array_sizeof.asp) | count() 的别名。 | 3 |
| [sort()](http://www.w3school.com.cn/php/func_array_sort.asp) | 对数组排序。 | 3 |
| [uasort()](http://www.w3school.com.cn/php/func_array_uasort.asp) | 使用用户自定义的比较函数对数组中的值进行排序并保持索引关联。 | 3 |
| [uksort()](http://www.w3school.com.cn/php/func_array_uksort.asp) | 使用用户自定义的比较函数对数组中的键名进行排序。 | 3 |
| [usort()](http://www.w3school.com.cn/php/func_array_usort.asp) | 使用用户自定义的比较函数对数组中的值进行排序。 | 3 |

**PHP Array 常量**

PHP：指示支持该常量的最早的 PHP 版本。

|  |  |  |
| --- | --- | --- |
| **常量** | **描述** | **PHP** |
| CASE\_LOWER | 用在 array\_change\_key\_case() 中将数组键名转换成小写字母。 |  |
| CASE\_UPPER | 用在 array\_change\_key\_case() 中将数组键名转换成大写字母。 |  |
| SORT\_ASC | 用在 array\_multisort() 函数中，使其升序排列。 |  |
| SORT\_DESC | 用在 array\_multisort() 函数中，使其降序排列。 |  |
| SORT\_REGULAR | 用于对对象进行通常比较。 |  |
| SORT\_NUMERIC | 用于对对象进行数值比较。 |  |
| SORT\_STRING | 用于对对象进行字符串比较。 |  |
| SORT\_LOCALE\_STRING | 基于当前区域来对对象进行字符串比较。 | 4 |
| COUNT\_NORMAL |  |  |
| COUNT\_RECURSIVE |  |  |
| EXTR\_OVERWRITE |  |  |
| EXTR\_SKIP |  |  |
| EXTR\_PREFIX\_SAME |  |  |
| EXTR\_PREFIX\_ALL |  |  |
| EXTR\_PREFIX\_INVALID |  |  |
| EXTR\_PREFIX\_IF\_EXISTS |  |  |
| EXTR\_IF\_EXISTS |  |  |
| EXTR\_REFS |  |  |

# array()

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array() 创建数组，带有键和值。如果在规定数组时省略了键，则生成一个整数键，这个 key 从 0 开始，然后以 1 进行递增。

要用 array() 创建一个关联数组，可使用 => 来分隔键和值。

要创建一个空数组，则不传递参数给 array()：

$new = array();

注意：array() 实际上是一种语言结构 (language construct)，通常用来定义直接量数组，但它的用法和函数的用法很相似，所以我们把它也列到手册中。

### 语法

array(key => value)

|  |  |
| --- | --- |
| **参数** | **描述** |
| key | 可选。规定 key，类型是数值或字符串。如果未设置，则生成整数类型的 key。 |
| value | 必需。规定值。 |

## 例子 1

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

print\_r($a);

?>

输出：

Array ( [a] => Dog [b] => Cat [c] => Horse )

## 例子 2

<?php

$a=array("Dog","Cat","Horse");

print\_r($a);

?>

输出：

Array ( [0] => Dog [1] => Cat [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_change\_key\_case() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_change\_key\_case() 函数将数组的所有的 KEY 都转换为大写或小写。

数组的数字索引不发生变化。如果未提供可选参数（即第二个参数），则默认转换为小写字母。

### 语法

array\_change\_key\_case(array,case)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |
| case | 可选。可能的值：   * CASE\_LOWER - 默认值。以小写字母返回数组的键。 * CASE\_UPPER - 以大写字母返回数组的键。 |

## 提示和注释：

注释：如果在运行该函数时两个或多个键相同，则最后的元素会覆盖其他元素（参见例子 2）。

## 例子 1

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

print\_r(array\_change\_key\_case($a,CASE\_UPPER));

?>

输出：

Array ( [A] => Cat [B] => Dog [C] => Horse )

## 例子 2

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse","B"=>"Bird");

print\_r(array\_change\_key\_case($a,CASE\_UPPER));

?>

输出：

Array ( [A] => Cat [B] => Bird [C] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_chunk() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_chunk() 函数把一个数组分割为新的数组块。

其中每个数组的单元数目由 size 参数决定。最后一个数组的单元数目可能会少几个。

可选参数 preserve\_key 是一个布尔值，它指定新数组的元素是否有和原数组相同的键（用于关联数组），还是从 0 开始的新数字键（用于索引数组）。默认是分配新的键。

### 语法

array\_chunk(array,size,preserve\_key)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |
| size | 必需。规定每个新数组包含多少个元素。 |
| preserve\_key | 可选。可能的值：   * true - 保留原始数组中的键名。 * false - 默认。每个结果数组使用从零开始的新数组索引。 |

## 例子 1

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse","d"=>"Cow");

print\_r(array\_chunk($a,2));

?>

输出：

Array (

[0] => Array ( [0] => Cat [1] => Dog )

[1] => Array ( [0] => Horse [1] => Cow )

)

## 例子 2

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse","d"=>"Cow");

print\_r(array\_chunk($a,2,true));

?>

输出：

Array (

[0] => Array ( [a] => Cat [b] => Dog )

[1] => Array ( [c] => Horse [d] => Cow )

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_combine() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_combine() 函数通过合并两个数组来创建一个新数组，其中的一个数组是键名，另一个数组的值为键值。

如果其中一个数组为空，或者两个数组的元素个数不同，则该函数返回 false。

### 语法

array\_combine(array1,array2)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。规定键名。 |
| array2 | 必需。规定值。 |

## 提示和注释

注释：两个参数必须有相同数目的元素。

## 例子

<?php

$a1=array("a","b","c","d");

$a2=array("Cat","Dog","Horse","Cow");

print\_r(array\_combine($a1,$a2));

?>

输出：

Array ( [a] => Cat [b] => Dog [c] => Horse [d] => Cow )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_count\_values() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_count\_values() 函数用于统计数组中所有值出现的次数。

本函数返回一个数组，其元素的键名是原数组的值，键值是该值在原数组中出现的次数。

### 语法

array\_count\_values(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |

## 例子

<?php

$a=array("Cat","Dog","Horse","Dog");

print\_r(array\_count\_values($a));

?>

输出：

Array ( [Cat] => 1 [Dog] => 2 [Horse] => 1 )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_diff() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_diff() 函数返回两个数组的差集数组。该数组包括了所有在被比较的数组中，但是不在任何其他参数数组中的键值。

在返回的数组中，键名保持不变。

### 语法

array\_diff(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。 |

## 提示和注释

提示：可用一个或任意多个数组与第一个数组进行比较。

注释：仅有值用于比较。

## 例子

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(3=>"Horse",4=>"Dog",5=>"Fish");

print\_r(array\_diff($a1,$a2));

?>

输出：

Array ( [0] => Cat )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_diff\_assoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_diff\_assoc() 函数返回两个数组的差集数组。该数组包括了所有在被比较的数组中，但是不在任何其他参数数组中的键和值。

和 [array\_diff() 函数](http://www.w3school.com.cn/php/func_array_diff.asp) 不同，本函数要求键名和键值都进行比较。返回的数组中键名保持不变。

### 语法

array\_diff\_assoc(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |

## 提示和注释

提示：可用一个或任意多个数组与第一个数组进行比较。

注释：键和值都用于比较。

## 例子

<?php

$a1=array(0=>"Cat",1=>"Dog";,2=>"Horse");

$a2=array(0=>"Rat",1=>"Horse";,2=>"Dog");

$a3=array(0=>"Horse",1=>"Dog",2=>"Cat");

print\_r(array\_diff\_assoc($a1,$a2,$a3));

?>

输出：

Array ( [0] => Cat [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_diff\_key() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_diff\_key() 函数返回一个数组，该数组包括了所有在被比较的数组中，但是不在任何其他参数数组中的键。

### 语法

array\_diff\_key(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |

## 提示和注释

提示：可用一个或任意多个数组与第一个数组进行比较。

注释：仅仅键名用于比较。

## 例子

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(2=>"Bird",3=>"Rat",4=>"Fish");

$a3=array(5=>"Horse",6=>"Dog",7=>"Bird");

print\_r(array\_diff\_key($a1,$a2,$a3));

?>

输出：

Array ( [0] => Cat [1] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_diff\_uassoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_diff\_uassoc() 函数使用用户自定义的回调函数 (callback) 做索引检查来计算两个或多个数组的差集。返回一个数组，该数组包括了在 *array1* 中但是不在任何其他参数数组中的值。

注意，与 [array\_diff() 函数](http://www.w3school.com.cn/php/func_array_diff.asp) 不同的是，键名也要进行比较。

参数 function 是用户自定义的用来比较两个数组的函数，该函数必须带有两个参数 - 即两个要进行对比的键名。因此与函数 [array\_diff\_assoc()](http://www.w3school.com.cn/php/func_array_diff_assoc.asp) 的行为正好相反，后者是用内部函数进行比较的。

返回的数组中键名保持不变。

### 语法

array\_diff\_uassoc(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |
| function | 必需。用户自定义函数的名称。 |

## 例子 1

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Dog",1=>"Cat",5=>"Horse");

print\_r(array\_diff\_uassoc($a1,$a2,"myfunction"));

?>

输出：

Array ( [0] => Dog [2] => Horse )

## 例子 2

如何为该函数分配多个数组：

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Dog",1=>"Cat",5=>"Horse");

$a3=array(6=>"Bird",0=>"Dog",5=>"Horse");

print\_r(array\_diff\_uassoc($a1,$a2,$a3,"myfunction"));

?>

输出：

Array ( [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_diff\_ukey() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_diff\_ukey() 返回一个数组，该数组包括了所有出现在 array1 中但是未出现在任何其它参数数组中的键名的值。注意关联关系保留不变。与 array\_diff() 不同的是，比较是根据键名而不是值来进行的。

此比较是通过用户提供的回调函数来进行的。如果认为第一个参数小于，等于，或大于第二个参数时必须分别返回一个小于零，等于零，或大于零的整数。

### 语法

array\_diff\_ukey(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |
| function | 必需。用户自定义函数的名称。 |

## 例子 1

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Rat",1=>"Bird",5=>"Monkey");

print\_r(array\_diff\_ukey($a1,$a2,"myfunction"));

?>

输出：

Array ( [0] => Dog [2] => Horse )

## 例子 2

如何为该函数分配多个数组：

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Rat",1=>"Bird",5=>"Monkey");

$a3=array(6=>"Dog",7=>"Donkey",0=>"Horse");

print\_r(array\_diff\_ukey($a1,$a2,$a3,"myfunction"));

?>

输出：

Array ( [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_fill() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_fill() 函数用给定的值填充数组，返回的数组有 number 个元素，值为 value。返回的数组使用数字索引，从 start 位置开始并递增。如果 number 为 0 或小于 0，就会出错。

### 语法

array\_fill(start,number,value)

|  |  |
| --- | --- |
| **参数** | **描述** |
| start | 必需。数值，规定键的起始索引。 |
| number | 必需。数值，规定填充的数量，其值必须大于 0。 |
| value | 必需。规定要插入的值。 |

## 例子

<?php

$a=array\_fill(2,3,"Dog");

print\_r($a);

?>

输出：

Array ( [2] => Dog [3] => Dog [4] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_filter() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_filter() 函数用回调函数过滤数组中的元素，如果自定义过滤函数返回 true，则被操作的数组的当前值就会被包含在返回的结果数组中， 并将结果组成一个新的数组。如果原数组是一个关联数组，键名保持不变。

### 语法

array\_filter(array,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |
| function | 必需。自定义函数的名称。 |

## 例子

<?php

function myfunction($v)

{

if ($v==="Horse")

{

return true;

}

return false;

}

$a=array(0=>"Dog",1=>"Cat",2=>"Horse");

print\_r(array\_filter($a,"myfunction"));

?>

输出：

Array ( [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_flip() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_flip() 函数返回一个反转后的数组，如果同一值出现了多次，则最后一个键名将作为它的值，所有其他的键名都将丢失。

如果原数组中的值的数据类型不是字符串或整数，函数将报错。

### 语法

array\_flip(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |

## 例子

<?php

$a=array(0=>"Dog",1=>"Cat",2=>"Horse");print\_r(array\_flip($a));

?>

输出：

Array ( [Dog] => 0 [Cat] => 1 [Horse] => 2 )

[PHP Array 函](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_intersect() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_intersect() 函数返回两个或多个数组的交集数组。

结果数组包含了所有在被比较数组中，也同时出现在所有其他参数数组中的值，键名保留不变。

注释：仅有值用于比较。

### 语法

array\_intersect(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |

## 例子

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(3=>"Horse",4=>"Dog",5=>"Fish");

print\_r(array\_intersect($a1,$a2));

?>

输出：

Array ( [1] => Dog [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_intersect\_assoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_intersect\_assoc() 函数返回两个或多个数组的交集数组。

与 [array\_intersect() 函数](http://www.w3school.com.cn/php/func_array_intersect.asp) 不同的是，本函数除了比较键值，还比较键名。返回的数组中元素的键名保持不变。

### 语法

array\_intersect\_assoc(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |

## 例子 1

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(3=>"Horse",1=>"Dog",0=>"Cat");

print\_r(array\_intersect\_assoc($a1,$a2));

?>

输出：

Array ( [0] => Cat [1] => Dog )

## 例子 2

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(3=>"Horse",1=>"Dog",5=>"Fish");

$a3=array(6=>"Cow",1=>"Dog",8=>"Fish");

print\_r(array\_intersect\_assoc($a1,$a2,$a3));

?>

输出：

Array ( [1] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_intersect\_key() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_intersect\_key() 函数使用键名比较计算数组的交集。

array\_intersect\_key() 返回一个数组，该数组包含了所有出现在被比较的数组中并同时出现在所有其它参数数组中的键名的值。

注释：仅有键名用于比较。

### 语法

array\_intersect\_key(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |

## 例子 1

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(2=>"Bird",0=>"Cat",4=>"Fish");

print\_r(array\_intersect\_key($a1,$a2));

?>

输出：

Array ( [0] => Cat [2] => Horse )

## 例子 2

<?php

$a1=array(0=>"Cat",1=>"Dog",2=>"Horse");

$a2=array(2=>"Bird",3=>"Rat",4=>"Fish");

$a3=array(2=>"Dog",6=>"Cow",7=>"Bird");

print\_r(array\_intersect\_key($a1,$a2,$a3));

?>

输出：

Array ( [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_intersect\_uassoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_intersect\_uassoc() 函数使用用户自定义的回调函数计算数组的交集，用回调函数比较索引。

array\_intersect\_uassoc() 返回一个数组，该数组包含了所有在 *array1* 中也同时出现在所有其它参数数组中的值。返回的数组中键名保持不变。

注意，与 [array\_intersect()](http://www.w3school.com.cn/php/func_array_intersect.asp) 不同的是除了比较键值，还要比较键名。

此比较是通过用户提供的回调函数来进行的。该函数带有两个参数，即两个要进行对比的键名。如果第一个参数小于第二个参数，则函数要返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个参数，则返回一个正数。

### 语法

array\_intersect\_uassoc(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |
| function | 用户自定义函数的名称。 |

## 例子 1

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Dog",1=>"Cat",5=>"Horse");

print\_r(array\_intersect\_uassoc($a1,$a2,"myfunction"));

?>

输出：

Array ( [1] => Cat )

## 例子 2

如何为函数分配多个数组：

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(0=>"Dog",1=>"Cat",5=>"Horse");

$a3=array(6=>"Bird",0=>"Dog",5=>"Horse");

print\_r(array\_intersect\_uassoc($a1,$a2,$a3,"myfunction"));

?>

输出：

Array ( [0] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_intersect\_ukey() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_intersect\_ukey() 函数用回调函数比较键名来计算数组的交集。

array\_intersect\_ukey() 返回一个数组，该数组包含了所有出现在 array1 中并同时出现在所有其它参数数组中的键名的值。

此比较是通过用户提供的回调函数来进行的。该函数带有两个参数，即两个要进行对比的键名。如果第一个参数小于第二个参数，则函数要返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个参数，则返回一个正数。

### 语法

array\_intersect\_uassoc(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。与其他数组进行比较的第一个数组。 |
| array2 | 必需。与第一个数组进行比较的数组。 |
| array3 | 可选。与第一个数组进行比较的数组。可以有多个。 |
| function | 用户自定义函数的名称。 |

## 例子 1

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(3=>"Rat",1=>"Bird",5=>"Monkey");

print\_r(array\_intersect\_ukey($a1,$a2,"myfunction"));

?>

输出：

Array ( [1] => Cat )

## 例子 2

如何为函数分配多个数组：

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1>$v2)

{

return 1;

}

else

{

return -1;

}

}

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse");

$a2=array(0=>"Rat",1=>"Bird",5=>"Monkey");

$a3=array(6=>"Dog",7=>"Donkey",0=>"Horse");

print\_r(array\_intersect\_ukey($a1,$a2,$a3,"myfunction"));

?>

输出：

Array ( [0] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_key\_exists() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_key\_exists() 函数判断某个数组中是否存在指定的 key，如果该 key 存在，则返回 true，否则返回 false。

### 语法

array\_key\_exists(key,array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| key | 必需。规定键名。 |
| array | 必需。规定输入的数组。 |

## 例子 1

<?php

$a=array("a"=>"Dog","b"=>"Cat");

if (array\_key\_exists("a",$a))

{

echo "Key exists!";

}

else

{

echo "Key does not exist!";

}

?>

输出：

Key exists!

## 例子 2

<?php

$a=array("a"=>"Dog","b"=>"Cat");

if (array\_key\_exists("c",$a))

{

echo "Key exists!";

}

else

{

echo "Key does not exist!";

}

?>

输出：

Key does not exist!

## 例子 2

<?php

$a=array("Dog",Cat");

if (array\_key\_exists(0,$a))

{

echo "Key exists!";

}

else

{

echo "Key does not exist!";

}

?>

输出：

Key exists!

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_keys() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_keys() 函数返回包含数组中所有键名的一个新数组。

如果提供了第二个参数，则只返回键值为该值的键名。

如果 strict 参数指定为 true，则 PHP 会使用全等比较 (===) 来严格检查键值的数据类型。

### 语法

array\_keys(array,value)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |
| value | 可选。指定值的索引（键）。 |
| strict | 可选。与 value 参数一起使用。可能的值：   * true - 根据类型返回带有指定值的键名。 * false - 默认值。不依赖类型。 |

## 例子 1

<?php

$a=array("a"=>"Horse","b"=>"Cat","c"=>"Dog");

print\_r(array\_keys($a));

?>

输出：

Array ( [0] => a [1] => b [2] => c )

## 例子 2

使用 value 参数：

<?php

$a=array("a"=>"Horse","b"=>"Cat","c"=>"Dog");

print\_r(array\_keys($a,"Dog"));

?>

输出：

Array ( [0] => c)

## 例子 3

使用 strict 参数 (false)：

<?php

$a=array(10,20,30,"10");

print\_r(array\_keys($a,"10",false));

?>

输出：

Array ( [0] => 0 [1] => 3 )

## 例子 4

使用 strict 参数 (true)：

<?php

$a=array(10,20,30,"10");

print\_r(array\_keys($a,"10",true));

?>

输出：

Array ( [0] => 3)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_map() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_map() 函数返回用户自定义函数作用后的数组。回调函数接受的参数数目应该和传递给 array\_map() 函数的数组数目一致。

### 语法

array\_map(function,array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| function | 必需。用户自定义函数的名称，或者是 null。 |
| array1 | 必需。规定数组。 |
| array2 | 可选。规定数组。 |
| array3 | 可选。规定数组。 |

## 例子 1

<?php

function myfunction($v)

{

if ($v==="Dog")

{

return "Fido";

}

return $v;

}

$a=array("Horse","Dog","Cat");

print\_r(array\_map("myfunction",$a));

?>

输出：

Array ( [0] => Horse [1] => Fido [2] => Cat )

## 例子 2

使用多个参数：

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return "same";

}

return "different";

}

$a1=array("Horse","Dog","Cat");

$a2=array("Cow","Dog","Rat");

print\_r(array\_map("myfunction",$a1,$a2));

?>

输出：

Array ( [0] => different [1] => same [2] => different )

## 例子 3

请看当自定义函数名设置为 null 时的情况：

<?php

$a1=array("Dog","Cat");

$a2=array("Puppy","Kitten");

print\_r(array\_map(null,$a1,$a2));

?>

输出：

Array (

[0] => Array ( [0] => Dog [1] => Puppy )

[1] => Array ( [0] => Cat [1] => Kitten )

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_merge() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_merge() 函数把两个或多个数组合并为一个数组。

如果键名有重复，该键的键值为最后一个键名对应的值（后面的覆盖前面的）。如果数组是数字索引的，则键名会以连续方式重新索引。

注释：如果仅仅向 array\_merge() 函数输入了一个数组，且键名是整数，则该函数将返回带有整数键名的新数组，其键名以 0 开始进行重新索引。（参见例子 2）

### 语法

array\_merge(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。输入的第一个数组。 |
| array2 | 必需。输入的第二个数组。 |
| array3 | 可选。可指定的多个输入数组。 |

## 例子 1

<?php

$a1=array("a"=>"Horse","b"=>"Dog");

$a2=array("c"=>"Cow","b"=>"Cat");

print\_r(array\_merge($a1,$a2));

?>

输出：

Array ( [a] => Horse [b] => Cat [c] => Cow )

## 例子 2

仅使用一个数组参数：

<?php

$a=array(3=>"Horse",4=>"Dog");

print\_r(array\_merge($a));

?>

输出：

Array ( [0] => Horse [1] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_merge\_recursive() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_merge\_recursive() 函数与 [array\_merge() 函数](http://www.w3school.com.cn/php/func_array_merge.asp) 一样，将一个或多个数组的元素的合并起来，一个数组中的值附加在前一个数组的后面。并返回作为结果的数组。

但是，与 [array\_merge()](http://www.w3school.com.cn/php/func_array_merge.asp) 不同的是，当有重复的键名时，值不会被覆盖，而是将多个相同键名的值递归组成一个数组。（参见例子 1）

### 语法

array\_merge\_recursive(array1,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。输入的第一个数组。 |
| array2 | 必需。输入的第二个数组。 |
| array3 | 可选。可指定的多个输入数组。 |

## 提示和注释

注释：当向 array\_merge\_recursive() 函数仅仅输入一个数组时，结果与 array\_merge() 相同。

## 例子 1

<?php

$a1=array("a"=>"Horse","b"=>"Dog");

$a2=array("c"=>"Cow","b"=>"Cat");

print\_r(array\_merge\_recursive($a1,$a2));

?>

输出：

Array (

[a] => Horse

[b] => Array ( [0] => Dog [1] => Cat )

[c] => Cow

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_multisort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_multisort() 函数对多个数组或多维数组进行排序。

参数中的数组被当成一个表的列并以行来进行排序 - 这类似 SQL 的 ORDER BY 子句的功能。第一个数组是要排序的主要数组。数组中的行（值）比较为相同的话，就会按照下一个输入数组中相应值的大小进行排序，依此类推。

第一个参数是数组，随后的每一个参数可能是数组，也可能是下面的排序顺序标志（排序标志用于更改默认的排列顺序）之一：

* SORT\_ASC - 默认，按升序排列。(A-Z)
* SORT\_DESC - 按降序排列。(Z-A)

随后可以指定排序的类型：

* SORT\_REGULAR - 默认。将每一项按常规顺序排列。
* SORT\_NUMERIC - 将每一项按数字顺序排列。
* SORT\_STRING - 将每一项按字母顺序排列。

### 语法

array\_multisort(array1,sorting order,sorting type,array2,array3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。规定输入的数组。 |
| sorting order | 可选。规定排列顺序。可能的值是 SORT\_ASC 和 SORT\_DESC。 |
| sorting type | 可选。规定排序类型。可能的值是SORT\_REGULAR、SORT\_NUMERIC和SORT\_STRING。 |
| array2 | 可选。规定输入的数组。 |
| array3 | 可选。规定输入的数组。 |

## 提示和注释

注释：字符串键名将被保留，但是数字键将被重新索引，从 0 开始，并以 1 递增。

注释：您可以在每个数组后设置排序顺序和排序类型。如果没有设置，每个数组参数会使用默认值。

## 例子 1

<?php

$a1=array("Dog","Cat");

$a2=array("Fido","Missy");

array\_multisort($a1,$a2);

print\_r($a1);

print\_r($a2);

?>

输出：

Array ( [0] => Cat [1] => Dog )

Array ( [0] => Missy [1] => Fido )

## 例子 2

当两个值相同时如何排序：

<?php

$a1=array("Dog","Dog","Cat");

$a2=array("Pluto","Fido","Missy");

array\_multisort($a1,$a2);

print\_r($a1);

print\_r($a2);

?>

输出：

Array ( [0] => Cat [1] => Dog [2] => Dog )

Array ( [0] => Missy [1] => Fido [2] => Pluto )

## 例子 3

带有排序参数：

<?php

$a1=array("Dog","Dog","Cat");

$a2=array("Pluto","Fido","Missy");

array\_multisort($a1,SORT\_ASC,$a2,SORT\_DESC);

print\_r($a1);

print\_r($a2);

?>

输出：

Array ( [0] => Cat [1] => Dog [2] => Dog )

Array ( [0] => Missy [1] => Pluto [2] => Fido )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_pad() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_pad() 函数向一个数组插入带有指定值的指定数量的元素。

### 语法

array\_pad(array,size,value)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定数组。 |
| size | 必需。指定的长度。整数则填补到右侧，负数则填补到左侧。 |
| value | 必需。用来填补的值。 |

## 提示和注释

提示：如何设置了负的长度值，该函数会在原始数组之前插入新的元素。（参见例子 2）

注释：如果 size 参数小于原始数组的长度，该函数不会删除任何元素。

## 例子 1

<?php

$a=array("Dog","Cat");

print\_r(array\_pad($a,5,0));

?>

输出：

Array ( [0] => Dog [1] => Cat [2] => 0 [3] => 0 [4] => 0 )

## 例子 2

带有负的 size 参数：

<?php

$a=array("Dog","Cat");

print\_r(array\_pad($a,-5,0));

?>

输出：

Array ( [0] => 0 [1] => 0 [2] => 0 [3] => Dog [4] => Cat )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_pop() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_pop() 函数删除数组中的最后一个元素。

### 语法

array\_pop(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组参数。 |

## 例子

<?php

$a=array("Dog","Cat","Horse");

array\_pop($a);

print\_r($a);

?>

输出：

Array ( [0] => Dog [1] => Cat )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_product() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_product() 函数计算并返回数组中所有值的乘积。

### 语法

array\_product(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组参数。 |

## 例子

<?php

$a=array(5,5);

echo(array\_product($a));

?>

输出：

25

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_push() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_push() 函数向第一个参数的数组尾部添加一个或多个元素（入栈），然后返回新数组的长度。

该函数等于多次调用 $array[] = $value。

### 语法

array\_push(array,value1,value2...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定一个数组。 |
| value1 | 必需。规定要添加的值。 |
| value2 | 可选。规定要添加的值。 |

## 提示和注释

注释：即使数组中有字符串键名，您添加的元素也始终是数字键。（参见例子 2）

注释：如果用 array\_push() 来给数组增加一个单元，还不如用 *$array[] =*，因为这样没有调用函数的额外负担。

注释：如果第一个参数不是数组，array\_push() 将发出一条警告。这和 $var[] 的行为不同，后者会新建一个数组。

## 例子 1

<?php

$a=array("Dog","Cat");

array\_push($a,"Horse","Bird");

print\_r($a);

?>

输出：

Array ( [0] => Dog [1] => Cat [2] => Horse [3] => Bird )

## 例子 2

带有字符串键的数组：

<?php

$a=array("a"=>"Dog","b"=>"Cat");

array\_push($a,"Horse","Bird");

print\_r($a);

?>

输出：

Array ( [a] => Dog [b] => Cat [0] => Horse [1] => Bird )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_rand() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_rand() 函数从数组中随机选出一个或多个元素，并返回。

第二个参数用来确定要选出几个元素。如果选出的元素不止一个，则返回包含随机键名的数组，否则返回该元素的键名。

注释：自 PHP 4.2.0 起，不再需要用 srand() 或 mt\_srand() 函数给随机数发生器播种，现已被自动完成。

### 语法

array\_rand(array,number)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组参数。 |
| number | 可选。默认是 1。规定返回多少个随机的元素。 |

## 例子 1

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

print\_r(array\_rand($a,1));

?>

输出：

b

## 例子 2

带有字符串键的数组：

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

print\_r(array\_rand($a,2));

?>

输出：

Array ( [0] => c [1] => b )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_reduce() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_reduce() 函数用回调函数迭代地将数组简化为单一的值。如果指定第三个参数，则该参数将被当成是数组中的第一个值来处理，或者如果数组为空的话就作为最终返回值。

### 语法

array\_reduce(array,function,initial)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |
| function | 必需。规定自定义回调函数的名称。 |
| initial | 可选。如果规定了该参数，该参数将作为数组中的第一个值来处理。 |

## 例子 1

<?php

function myfunction($v1,$v2)

{

return $v1 . "-" . $v2;

}

$a=array("Dog","Cat","Horse");

print\_r(array\_reduce($a,"myfunction"));

?>

输出：

-Dog-Cat-Horse

## 例子 2

带有 initial 参数：

<?php

function myfunction($v1,$v2)

{

return $v1 . "-" . $v2;

}

$a=array("Dog","Cat","Horse");

print\_r(array\_reduce($a,"myfunction",5));

?>

输出：

5-Dog-Cat-Horse

## 例子 3

返回总和：

<?php

function myfunction($v1,$v2)

{

return $v1+$v2;

}

$a=array(10,15,20);

print\_r(array\_reduce($a,"myfunction",5));

?>

输出：

50

# array\_reverse() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_reverse() 函数将原数组中的元素顺序翻转，创建新的数组并返回。如果第二个参数指定为 true，则元素的键名保持不变，否则键名将丢失。

### 语法

array\_reverse(array,preserve)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定数组。 |
| preserve | 可选。规定是否保留原始数组的键名。可能的值：true 和 false。  这个参数是 PHP 4.0.3 中新加的。 |

## 例子

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

print\_r(array\_reverse($a));

?>

输出：

Array ( [c] => Horse [b] => Cat [a] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_search() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_search() 函数与 [in\_array()](http://www.w3school.com.cn/php/func_array_in_array.asp) 一样，在数组中查找一个键值。如果找到了该值，匹配元素的键名会被返回。如果没找到，则返回 false。

在 PHP 4.2.0 之前，函数在失败时返回 null 而不是 false。

如果第三个参数 strict 被指定为 true，则只有在数据类型和值都一致时才返回相应元素的键名。

### 语法

array\_search(value,array,strict)

|  |  |
| --- | --- |
| **参数** | **描述** |
| value | 必需。规定在数组中搜索的值。 |
| array | 必需。被搜索的数组。 |
| strict | 可选。可能的值：   * true * false - 默认   如果值设置为 true，还将在数组中检查给定值的类型。（参见例子 2） |

## 例子 1

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

echo array\_search("Dog",$a);

?>

输出：

a

## 例子 2

<?php

$a=array("a"=>"5","b"=>5,"c"=>"5");

echo array\_search(5,$a,true);

?>

输出：

b

[PHP Array 函](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_shift() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_shift() 函数删除数组中的第一个元素，并返回被删除元素的值。

注释：如果键是数字的，所有元素都将获得新的键，从 0 开始，并以 1 递增。（参见例子 2）。

### 语法

array\_shift(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |

## 例子 1

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse");

echo array\_shift($a);

print\_r ($a);

?>

输出：

Dog

Array ( [b] => Cat [c] => Horse )

## 例子 2

带有数字键：

<?php

$a=array(0=>"Dog",1=>"Cat",2=>"Horse");

echo array\_shift($a);

print\_r ($a);

?>

输出：

Dog

Array ( [0] => Cat [1] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_slice() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_slice() 函数在数组中根据条件取出一段值，并返回。

注释：如果数组有字符串键，所返回的数组将保留键名。（参见例子 4）

### 语法

array\_slice(array,offset,length,preserve)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |
| offset | 必需。数值。规定取出元素的开始位置。  如果是正数，则从前往后开始取，如果是负值，从后向前取 offset 绝对值。 |
| length | 可选。数值。规定被返回数组的长度。  如果是负数，则从后向前，选取该值绝对值数目的元素。如果未设置该值，则返回所有元素。 |
| preserve | 可选。可能的值：   * true - 保留键 * false - 默认 - 重置键 |

## 例子 1

<?php

$a=array(0=>"Dog",1=>"Cat",2=>"Horse",3=>"Bird");

print\_r(array\_slice($a,1,2));

?>

输出：

Array ( [0] => Cat [1] => Horse )

## 例子 2

带有负的 offset 参数：

<?php

$a=array(0=>"Dog",1=>"Cat",2=>"Horse",3=>"Bird");

print\_r(array\_slice($a,-2,1));

?>

输出：

Array ( [0] => Horse )

## 例子 3

preserve 参数设置为 true：

<?php

$a=array(0=>"Dog",1=>"Cat",2=>"Horse",3=>"Bird");

print\_r(array\_slice($a,1,2,true));

?>

输出：

Array ( [1] => Cat [2] => Horse )

## 例子 4

带有字符串键：

<?php

$a=array("a"=>"Dog","b"=>"Cat","c"=>"Horse","d"=>"Bird");

print\_r(array\_slice($a,1,2));

?>

输出：

Array ( [b] => Cat [c] => Horse )

# array\_splice() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_splice() 函数与 array\_slice() 函数类似，选择数组中的一系列元素，但不返回，而是删除它们并用其它值代替。

如果提供了第四个参数，则之前选中的那些元素将被第四个参数指定的数组取代。

最后生成的数组将会返回。

### 语法

array\_splice(array,offset,length,array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定数组。 |
| offset | 必需。数值。如果 offset 为正，则从输入数组中该值指定的偏移量开始移除。如果 offset 为负，则从输入数组末尾倒数该值指定的偏移量开始移除。 |
| length | 可选。数值。如果省略该参数，则移除数组中从 offset 到 结尾的所有部分。如果指定了 length 并且为正值，则移除这么多元素。如果指定了 length 且为负值，则移除从 offset 到数组末尾倒数 length 为止中间所有的元素。 |
| array | 被移除的元素由此数组中的元素替代。如果没有移除任何值，则此数组中的元素将插入到指定位置。 |

## 提示和注释

提示：如果函数没有删除任何元素 (length=0)，则替代数组将从 start 参数的位置插入。（参见例子 3）

注释：不保留替代数组中的键。

## 例子 1

<?php

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse",3=>"Bird");

$a2=array(0=>"Tiger",1=>"Lion");

array\_splice($a1,0,2,$a2);

print\_r($a1);

?>

输出：

Array ( [0] => Tiger [1] => Lion [2] => Horse [3] => Bird )

## 例子 2

与例子 1 相同，但是输出返回的数组：

<?php

$a1=array(0=>"Dog",1=>"Cat",2=>"Horse",3=>"Bird");

$a2=array(0=>"Tiger",1=>"Lion");

print\_r(array\_splice($a1,0,2,$a2));

?>

输出：

Array ( [0] => Dog [1] => Cat )

## 例子 3

length 参数设置为 0：

<?php

$a1=array(0=>"Dog",1=>"Cat");

$a2=array(0=>"Tiger",1=>"Lion");

array\_splice($a1,1,0,$a2);

print\_r($a1);

?>

输出：

Array ( [0] => Dog [1] => Tiger [2] => Lion [3] => Cat )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_sum() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_sum() 函数返回数组中所有值的总和。

如果所有值多是整数，则返回一个整数值。如果其中有一个或多个值是浮点数，则返回浮点数。

PHP 4.2.1 之前的版本修改了传入的数组本身，将其中的字符串值转换成数值（大多数情况下都转换成了零，根据具体制而定）。

### 语法

array\_sum(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |

## 例子

<?php

$a=array(0=>"5",1=>"15",2=>"25");

echo array\_sum($a);

?>

输出：

45

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_udiff() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_udiff() 函数返回一个数组，该数组包括了所有在被比较数组中，但是不在任何其它参数数组中的值，键名保留不变。

array\_udiff() 函数与 [array\_diff() 函数](http://www.w3school.com.cn/php/func_array_diff.asp) 的行为不同，后者用内部函数进行比较。

数据的比较是用 array\_udiff() 函数的 *function* 进行的。*function* 函数带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

### 语法

array\_udiff(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function | 可选。自定义的比较回调函数。 |

## 注释和提示

注释：该函数只进行键值的比较，不比较键名。如 "a"=>1 和 "b"=>1 这两个元素视作相等的。

注释：array\_udiff() 函数仅检查多维数组中的一维。

## 例子

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array(1=>"Cat",2=>"Dog",3=>"Fish");

print\_r(array\_udiff($a1,$a2,"myfunction"));

?>

输出：

Array ( [c] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_udiff\_assoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_udiff\_assoc() 函数返回 *array1* 中存在但其它数组中都不存在的部分。

注意与 [array\_diff()](http://www.w3school.com.cn/php/func_array_diff.asp) 以及 [array\_udiff()](http://www.w3school.com.cn/php/func_array_udiff.asp) 不同的是键名也用于比较。同时进行键名和键值的比较。如 "a"=>1 和 "b"=>1 这两个元素是不相等的。

数组数据的比较是用用户提供的回调函数进行的。在此方面和 [array\_diff\_assoc()](http://www.w3school.com.cn/php/func_array_diff_assoc.asp) 的行为正好相反，后者是用内部函数进行比较的。

array\_udiff\_assoc() 函数的 *function* 参数指定的函数用于比较元素是否相等。*function* 函数带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

### 语法

array\_udiff\_assoc(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function | 可选。自定义的比较回调函数。 |

## 例子

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array("a"=>"Cat","b"=>"Horse","c"=>"Dog");

print\_r(array\_udiff\_assoc($a1,$a2,"myfunction"));

?>

输出：

Array ( [b] => Dog [c] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_udiff\_uassoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_udiff\_uassoc() 函数返回 *array1* 数组中存在但其它数组中都不存在的部分。返回的数组中键名保持不变。

注意与 [array\_diff()](http://www.w3school.com.cn/php/func_array_diff.asp) 以及 [array\_udiff()](http://www.w3school.com.cn/php/func_array_udiff.asp) 不同的是键名也用于比较。同时进行键名和键值的比较，如 "a"=>1 和 "b"=>1 这两个元素是不相等的。

对键名（索引）的检查也是由回调函数 *function1* 进行的。这和 [array\_udiff\_assoc()](http://www.w3school.com.cn/php/func_array_udiff_assoc.asp) 的行为不同，后者是用内部函数比较索引的。

数组数据的比较是使用用户提供的回调函数 *function2* 进行的。在此方面和 [array\_diff\_assoc()](http://www.w3school.com.cn/php/func_array_diff_assoc.asp) 的行为正好相反，后者是用内部函数进行比较的。

这两个函数都带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

### 语法

array\_udiff\_uassoc(array1,array2,array3...,function1,function2)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function1 | 必需。比较键名的自定义函数。 |
| function2 | 必需。比较值的自定义函数。 |

注释：*function1* 指定的函数用于比较键名是否相等。*function2* 指定的函数用于比较键值是否相等。

## 例子

<?php

function myfunction\_key($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

function myfunction\_value($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array("a"=>"Cat","b"=>"Dog","c"=>"Fish");

print\_r(array\_udiff\_uassoc($a1,$a2,"myfunction\_key","myfunction\_value"));

?>

输出：

Array ( [c] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_uintersect() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_uintersect() 函数计算数组的交集，用回调函数比较数据。

array\_uintersect() 返回一个数组，该数组包含了所有在 *array1* 中也同时出现在所有其它参数数组中的值。数据（键值）比较是用回调函数进行的。

### 语法

array\_uintersect(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function | 必需。自定义函数的名称。 |

### 说明

使用用户自定义的回调函数 *function* 来计算两个或多个数组的交集（即在 *array1* 中存在同时也在其它任何数组中存在的所有数组元素），并返回结果数组。

只进行键值的比较，不比较键名，如 "a"=>1 和 "b"=>1 这两个元素视作相等的。

*function* 参数指定的函数用于比较元素是否相等。function 函数带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

返回的数组中键名保持不变。

## 例子

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

if ($v1 > $v2) return 1;

{

return -1;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array(1=>"Cat",2=>"Dog",3=>"Fish");

print\_r(array\_uintersect($a1,$a2,"myfunction"));

?>

输出：

Array ( [a] => Cat [b] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_uintersect\_assoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_uintersect\_assoc() 函数带索引检查计算数组的交集，用回调函数比较数据。

array\_uintersect\_assoc() 返回一个数组，该数组包含了所有在 *array1* 中也同时出现在所有其它参数数组中的值。

注意，与 [array\_uintersect()](http://www.w3school.com.cn/php/func_array_uintersect.asp) 不同的是键名也要比较。数据（键值）是用回调函数比较的。

### 语法

array\_uintersect\_assoc(array1,array2,array3...,function)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function | 必需。自定义函数的名称。 |

### 说明

使用用户自定义的回调函数 *function* 来计算两个或多个数组的交集（即在 *array1* 中存在，同时也在其它任何数组中存在的所有数组元素），并返回结果数组。

同时进行键名和键值的比较，如 "a"=>1 和 "b"=>1 这两个元素是不相等的。

*function* 参数指定的函数用于比较元素是否相等。*function* 函数带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

返回的数组中键名保持不变。

## 例子

<?php

function myfunction($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array("a"=>"Cat","b"=>"Horse","c"=>"Dog");

print\_r(array\_uintersect\_assoc($a1,$a2,"myfunction"));

?>

输出：

Array ( [a] => Cat )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_uintersect\_uassoc() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_uintersect\_uassoc 函数带索引检查计算数组的交集，用回调函数来比较数据和索引。

array\_uintersect\_uassoc() 返回一个数组，该数组包含了所有在 *array1* 中也同时出现在所有其它参数数组中的值。

注意，与 [array\_uintersect()](http://www.w3school.com.cn/php/func_array_uintersect.asp) 不同的是键名也要比较。键值和键名（索引）都是用回调函数比较的。

### 语法

array\_uintersect\_uassoc(array1,array2,array3...,function1,function2)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array1 | 必需。被比较的数组。 |
| array2 | 必需。用来做比较的数组。 |
| array3 | 可选。用来做比较的数组，可有多个。 |
| function1 | 必需。比较键名的自定义函数。 |
| function2 | 必需。比较键值的自定义函数。 |

### 说明

使用用户自定义的回调函数 *function1* 和 *function2* 来计算两个或多个数组的交集（即在 *array1* 中存在，同时也在其它任何数组中存在的所有数组元素），并返回结果数组。

同时进行键名和键值的比较，如 "a"=>1 和 "b"=>1 这两个元素是不相等的。

*function1* 指定的函数用于比较键名是否相等。*function2* 指定的函数用于比较键值是否相等。这两个函数都带有两个将进行比较的参数。如果第一个参数小于第二个参数，则函数返回一个负数，如果两个参数相等，则要返回 0，如果第一个参数大于第二个，则返回一个正数。

返回的数组中键名保持不变。

## 例子

<?php

function myfunction\_key($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

function myfunction\_value($v1,$v2)

{

if ($v1===$v2)

{

return 0;

}

return 1;

}

$a1=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

$a2=array("a"=>"Cat","b"=>"Dog","c"=>"Dog");

print\_r(array\_uintersect\_uassoc($a1,$a2,"myfunction\_key","myfunction\_value"));

?>

输出：

Array ( [a] => Cat [b] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_unique() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_unique() 函数移除数组中的重复的值，并返回结果数组。

当几个数组元素的值相等时，只保留第一个元素，其他的元素被删除。

返回的数组中键名不变。

### 语法

array\_unique(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |

### 说明

array\_unique() 先将值作为字符串排序，然后对每个值只保留第一个遇到的键名，接着忽略所有后面的键名。这并不意味着在未排序的 *array* 中同一个值的第一个出现的键名会被保留。

## 提示和注释

注释：被返回的数组将保持第一个数组元素的键类型。

## 例子

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Cat");

print\_r(array\_unique($a));

?>

输出：

Array ( [a] => Cat [b] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_unshift() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_unshift() 函数在数组开头插入一个或多个元素。

被加上的元素作为一个整体添加，这些元素在数组中的顺序和在参数中的顺序一样。

该函数会返回数组中元素的个数。

### 语法

array\_unshift(array,value1,value2,value3...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定输入的数组。 |
| value1 | 必需。规定插入的值。 |
| value2 | 可选。规定插入的值。 |
| value3 | 可选。规定插入的值。 |

## 提示和注释

注释：所有的数值键名将修改为从零开始重新计数，所有的字符串键名保持不变。

## 例子 1

<?php

$a=array("a"=>"Cat","b"=>"Dog");

array\_unshift($a,"Horse");

print\_r($a);

?>

输出：

Array ( [0] => Horse [a] => Cat [b] => Dog )

## 例子 2

返回键值：

<?php

$a=array("a"=>"Cat","b"=>"Dog");

print\_r(array\_unshift($a,"Horse"));

?>

输出：

3

## 例子 3

数组带有数值键：

<?php

$a=array(0=>"Cat",1=>"Dog");

array\_unshift($a,"Horse");

print\_r($a);

?>

输出：

Array ( [0] => Horse [1] => Cat [2] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_values() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_values() 函数返回一个包含给定数组中所有键值的数组，但不保留键名。

### 语法

array\_values(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定给定的数组。 |

## 提示和注释

提示：被返回的数组将使用数值键，从 0 开始且以 1 递增。

## 例子

<?php

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

print\_r(array\_values($a));

?>

输出：

Array ( [0] => Cat [1] => Dog [2] => Horse )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_walk() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

array\_walk() 函数对数组中的每个元素应用回调函数。如果成功则返回 TRUE，否则返回 FALSE。

典型情况下 *function* 接受两个参数。*array* 参数的值作为第一个，键名作为第二个。如果提供了可选参数 *userdata* ，将被作为第三个参数传递给回调函数。

如果 *function* 函数需要的参数比给出的多，则每次 array\_walk() 调用 *function* 时都会产生一个 E\_WARNING 级的错误。这些警告可以通过在 array\_walk() 调用前加上 PHP 的错误操作符 @ 来抑制，或者用 [error\_reporting()](http://www.w3school.com.cn/php/func_error_reporting.asp)。

### 语法

array\_walk(array,function,userdata...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定数组。 |
| function | 必需。用户自定义函数的名称。 |
| userdata | 可选。用户输入的值，可作为回调函数的参数。 |

## 提示和注释

提示：您可以为函数设置一个或多个参数。

注释：如果回调函数需要直接作用于数组中的值，可以将回调函数的第一个参数指定为引用：&$value。（参见例子 3）

注释：将键名和 userdata 传递到 function 中是 PHP 4.0 新增加的。

## 例子 1

<?php

function myfunction($value,$key)

{

echo "The key $key has the value $value<br />";

}

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

array\_walk($a,"myfunction");

?>

输出：

The key a has the value Cat

The key b has the value Dog

The key c has the value Horse

## 例子 2

带有一个参数：

<?php

function myfunction($value,$key,$p)

{

echo "$key $p $value<br />";

}

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

array\_walk($a,"myfunction","has the value");

?>

输出：

a has the value Cat

b has the value Dog

c has the value Horse

## 例子 3

改变数组元素的值（请注意 &$value）：

<?php

function myfunction(&$value,$key)

{

$value="Bird;

}

$a=array("a"=>"Cat","b"=>"Dog","c"=>"Horse");

array\_walk($a,"myfunction");

print\_r($a);

?>

输出：

Array ( [a] => Bird [b] => Bird [c] => Bird )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# array\_walk\_recursive() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

与 [array\_walk() 函数](http://www.w3school.com.cn/php/func_array_walk.asp) 类似，array\_walk\_recursive() 函数对数组中的每个元素应用回调函数。不一样的是，如果原数组中的元素也是数组，就会递归地调用回调函数，也就是说，会递归到更深层的数组中去。

典型情况下，*function* 接受两个参数。*array* 参数的值作为第一个，键名作为第二个。如果提供了可选参数 *userdata* ，将被作为第三个参数传递给回调函数。

如果回调函数需要直接作用于数组中的值，可以将回调函数的第一个参数指定为引用，这样对这些单元的任何改变也将会改变原始数组本身。

### 语法

array\_walk\_recursive(array,function,userdata)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定数组。 |
| function | 必需。用户自定义函数的名称。 |
| userdata | 可选。用户输入的值，可作为回调函数的参数。 |

## 例子

<?php

function myfunction($value,$key)

{

echo "The key $key has the value $value<br />";

}

$a1=array("a"=>"Cat","b"=>"Dog");

$a2=array($a1,"1"=>"Bird","2"=>"Horse");

array\_walk\_recursive($a2,"myfunction");

?>

输出：

The key a has the value Cat

The key b has the value Dog

The key 1 has the value Bird

The key 2 has the value Horse

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# arsort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

arsort() 函数对数组进行逆向排序并保持索引关系。主要用于对那些单元顺序很重要的结合数组进行排序。

可选的第二个参数包含了附加的排序标识。

如果成功则返回 TRUE，否则返回 FALSE。

### 语法

arsort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。输入的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

arsort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

[c] => Horse

[a] => Dog

[b] => Cat

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# asort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

asort() 函数对数组进行排序并保持索引关系。主要用于对那些单元顺序很重要的结合数组进行排序。

可选的第二个参数包含了附加的排序标识。

如果成功则返回 TRUE，否则返回 FALSE。

### 语法

asort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。输入的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

asort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

[b] => Cat

[a] => Dog

[c] => Horse

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# compact() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

compact() 函数创建一个由参数所带变量组成的数组。如果参数中存在数组，该数组中变量的值也会被获取。

本函数返回的数组是一个关联数组，键名为函数的参数，键值为参数中变量的值。

本函数执行的行为与 [extract()](http://www.w3school.com.cn/php/func_array_extract.asp) 正好相反。

### 语法

compact(var1,var2...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| var1 | 必需。可以是带有变量名的字符串，或者是一个变量数组。 |
| var2 | 可选。可以是带有变量名的字符串，或者是一个变量数组。允许多个参数。 |

## 注释和提示

注释：任何没有变量名与之对应的字符串都被略过。

## 例子 1

<?php

$firstname = "Peter";

$lastname = "Griffin";

$age = "38";

$result = compact("firstname", "lastname", "age");

print\_r($result);

?>

输出：

Array

(

[firstname] => Peter

[lastname] => Griffin

[age] => 38

)

## 例子 2

使用没有对应变量名的字符串，以及一个变量名数组：

<?php

$firstname = "Peter";

$lastname = "Griffin";

$age = "38";

$name = array("firstname", "lastname");

$result = compact($name, "location", "age");

print\_r($result);

?>

输出：

Array

(

[firstname] => Peter

[lastname] => Griffin

[age] => 38

)

# count() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

count() 函数计算数组中的单元数目或对象中的属性个数。

对于数组，返回其元素的个数，对于其他值，返回 1。如果参数是变量而变量没有定义，则返回 0。如果 mode 被设置为 COUNT\_RECURSIVE（或 1），则会递归底计算多维数组中的数组的元素个数。

### 语法

count(array,mode)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要计数的数组或对象。 |
| mode | 可选。规定函数的模式。可能的值：   * 0 - 默认。不检测多维数组（数组中的数组）。 * 1 - 检测多维数组。   注释：该参数是 PHP 4.2 中加入的。 |

## 提示和注释

注释：当变量未被设置，或是变量包含一个空的数组，该函数会返回 0。可使用 isset() 变量来测试变量是否被设置。

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

$result = count($people);

echo $result;

?>

输出：

4

# [PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)current() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

current() 函数返回数组中的当前元素（单元）。

每个数组中都有一个内部的指针指向它“当前的”元素，初始指向插入到数组中的第一个元素。

current() 函数返回当前被内部指针指向的数组元素的值，并不移动指针。如果内部指针指向超出了单元列表的末端，current() 返回 FALSE。

### 语法

current(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 提示和注释

注释：如果有空的元素，或元素没有值，该函数也返回 FALSE。

提示：该函数不会移动内部指针。要做到这一点，请使用 [next()](http://www.w3school.com.cn/php/func_array_next.asp) 和 [prev()](http://www.w3school.com.cn/php/func_array_prev.asp) 函数。

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo current($people) . "<br />";

?>

输出：

Peter

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# each() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

each() 函数生成一个由数组当前内部指针所指向的元素的键名和键值组成的数组，并把内部指针向前移动。

返回的数组中包括的四个元素：键名为 0，1，key 和 value。单元 0 和 key 包含有数组单元的键名，1 和 value 包含有数据。

如果内部指针越过了数组范围，本函数将返回 FALSE。

### 语法

each(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子 1

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

print\_r (each($people));

?>

输出：

Array ( [1] => Peter [value] => Peter [0] => 0 [key] => 0 )

## 例子 2

each() 经常和 list() 结合使用来遍历数组。本例与上例类似，不过循环输出了整个数组：

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

reset($people);

while (list($key, $val) = each($people))

{

echo "$key => $val<br />";

}

?>

输出：

0 => Peter

1 => Joe

2 => Glenn

3 => Cleveland

### 例子解释

因为将一个数组赋值给另一个数组时会重置原来的数组指针，因此在上例中如果我们在循环内部将 $people 赋给了另一个变量的话将会导致无限循环。

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# each() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

end() 函数将数组内部指针指向最后一个元素，并返回该元素的值（如果成功）。

### 语法

end(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo current($people) . "<br />";

echo end($people);

?>

输出：

Peter

Cleveland

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# extract() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

PHP extract() 函数从数组中把变量导入到当前的符号表中。

对于数组中的每个元素，键名用于变量名，键值用于变量值。

第二个参数 type 用于指定当某个变量已经存在，而数组中又有同名元素时，extract() 函数如何对待这样的冲突。

本函数返回成功设置的变量数目。

### 语法

extract(array,extract\_rules,prefix)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的输入。 |
| extract\_rules | 可选。extract() 函数将检查每个键名是否为合法的变量名，同时也检查和符号表中的变量名是否冲突。  对非法、数字和冲突的键名的处理将根据此参数决定。可以是以下值之一：  可能的值：   * EXTR\_OVERWRITE - 默认。如果有冲突，则覆盖已有的变量。 * EXTR\_SKIP - 如果有冲突，不覆盖已有的变量。（忽略数组中同名的元素） * EXTR\_PREFIX\_SAME - 如果有冲突，在变量名前加上前缀 prefix。自 PHP 4.0.5 起，这也包括了对数字索引的处理。 * EXTR\_PREFIX\_ALL - 给所有变量名加上前缀 prefix（第三个参数）。 * EXTR\_PREFIX\_INVALID - 仅在非法或数字变量名前加上前缀 prefix。本标记是 PHP 4.0.5 新加的。 * EXTR\_IF\_EXISTS - 仅在当前符号表中已有同名变量时，覆盖它们的值。其它的都不处理。可以用在已经定义了一组合法的变量，然后要从一个数组例如 $\_REQUEST 中提取值覆盖这些变量的场合。本标记是 PHP 4.2.0 新加的。 * EXTR\_PREFIX\_IF\_EXISTS - 仅在当前符号表中已有同名变量时，建立附加了前缀的变量名，其它的都不处理。本标记是 PHP 4.2.0 新加的。 * EXTR\_REFS - 将变量作为引用提取。这有力地表明了导入的变量仍然引用了 var\_array 参数的值。可以单独使用这个标志或者在 extract\_type 中用 OR 与其它任何标志结合使用。本标记是 PHP 4.3.0 新加的。 |
| prefix | 可选。请注意 prefix 仅在 extract\_type 的值是 EXTR\_PREFIX\_SAME，EXTR\_PREFIX\_ALL，EXTR\_PREFIX\_INVALID 或 EXTR\_PREFIX\_IF\_EXISTS 时需要。如果附加了前缀后的结果不是合法的变量名，将不会导入到符号表中。  前缀和数组键名之间会自动加上一个下划线。 |

## 例子 1

<?php

$a = 'Original';

$my\_array = array("a" => "Cat","b" => "Dog", "c" => "Horse");

extract($my\_array);

echo "\$a = $a; \$b = $b; \$c = $c";

?>

输出：

$a = Cat; $b = Dog; $c = Horse

## 例子 2

使用全部参数：

<?php

$a = 'Original';

$my\_array = array("a" => "Cat","b" => "Dog", "c" => "Horse");

extract($my\_array, EXTR\_PREFIX\_SAME, 'dup');

echo "\$a = $a; \$b = $b; \$c = $c; \$dup\_a = $dup\_a;";

?>

输出：

$a = Original; $b = Dog; $c = Horse; $dup\_a = Cat;

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# in\_array() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

in\_array() 函数在数组中搜索给定的值。

### 语法

in\_array(value,array,type)

|  |  |
| --- | --- |
| **参数** | **描述** |
| value | 必需。规定要在数组搜索的值。 |
| array | 必需。规定要搜索的数组。 |
| type | 可选。如果设置该参数为 true，则检查搜索的数据与数组的值的类型是否相同。 |

### 说明

如果给定的值 *value* 存在于数组 *array* 中则返回 true。如果第三个参数设置为 true，函数只有在元素存在于数组中且数据类型与给定值相同时才返回 true。

如果没有在数组中找到参数，函数返回 false。

注释：如果 *value* 参数是字符串，且 *type* 参数设置为 true，则搜索区分大小写。

## 例子 1

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

if (in\_array("Glenn",$people))

{

echo "Match found";

}

else

{

echo "Match not found";

}

?>

输出：

Match found

## 例子 2

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland", 23);

if (in\_array("23",$people, TRUE))

{

echo "Match found<br />";

}

else

{

echo "Match not found<br />";

}if (in\_array("Glenn",$people, TRUE))

{

echo "Match found<br />";

}

else

{

echo "Match not found<br />";

}if (in\_array(23,$people, TRUE))

{

echo "Match found<br />";

}

else

{

echo "Match not found<br />";

}

?>

输出：

Match not found

Match found

Match found

# key() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

key() 函数返回数组内部指针当前指向元素的键名。

若失败，则返回 FALSE。

该函数与 [current()](http://www.w3school.com.cn/php/func_array_current.asp) 类似，只是返回的结果不同。current() 函数返回的是元素的值，而 key() 函数返回的是元素的键名。

### 语法

key(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo "The key from the current position is: " . key($people);

?>

输出：

The key from the current position is: 0

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# krsort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

krsort() 函数将数组按照键逆向排序，为数组值保留原来的键。

可选的第二个参数包含附加的排序标志。

若成功，则返回 TRUE，否则返回 FALSE。

### 语法

krsort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要排序的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

krsort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

[c] => Horse

[b] => Cat

[a] => Dog

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# ksort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

ksort() 函数按照键名对数组排序，为数组值保留原来的键。

可选的第二个参数包含附加的排序标志。

若成功，则返回 TRUE，否则返回 FALSE。

### 语法

ksort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要排序的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

ksort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

[a] => Dog

[b] => Cat

[c] => Horse

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# list() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

list() 函数用数组中的元素为一组变量赋值。

注意，与 [array()](http://www.w3school.com.cn/php/func_array.asp) 类似，list() 实际上是一种语言结构，不是函数。

### 语法

list(var1,var2...)

|  |  |
| --- | --- |
| **参数** | **描述** |
| var1 | 必需。第一个需要赋值的变量。 |
| var2 | 可选。可以有多个变量。 |

## 提示和注释

注释：该函数只用于数字索引的数组，且假定数字索引从 0 开始。

## 例子 1

<?php

$my\_array = array("Dog","Cat","Horse");

list($a, $b, $c) = $my\_array;

echo "I have several animals, a $a, a $b and a $c.";

?>

输出：

I have several animals, a Dog, a Cat and a Horse.

## 例子 2

<?php

$my\_array = array("Dog","Cat","Horse");

list($a, , $c) = $my\_array;

echo "Here I only use the $a and $c variables.";

?>

输出：

Here I only use the Dog and Horse variables.

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# natcasesort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

natcasesort() 函数用不区分大小写的自然顺序算法对给定数组中的元素排序。

natcasesort() 函数实现了“自然排序”，即数字从 1 到 9 的排序方法，字母从 a 到 z 的排序方法，短者优先，该函数不区分大小写。数组的索引与单元值保持关联。

如果成功，则该函数返回 TRUE，否则返回 FALSE。

### 语法

natcasesort(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要进行排序的数组。 |

## 提示和注释

提示：natcasesort() 是 [natsort() 函数](http://www.w3school.com.cn/php/func_array_natsort.asp) 的不区分大小写字母的版本。

## 例子

<?php

$temp\_files = array("temp15.txt","Temp10.txt",

"temp1.txt","Temp22.txt","temp2.txt");

natsort($temp\_files);

echo "Natural order: ";

print\_r($temp\_files);

echo "<br />";

natcasesort($temp\_files);

echo "Natural order case insensitve: ";

print\_r($temp\_files);

?>

输出：

Natural order:

Array

(

[0] => Temp10.txt

[1] => Temp22.txt

[2] => temp1.txt

[4] => temp2.txt

[3] => temp15.txt

)

Natural order case insensitve:

Array

(

[2] => temp1.txt

[4] => temp2.txt

[0] => Temp10.txt

[3] => temp15.txt

[1] => Temp22.txt

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# natsort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

natsort() 函数用自然顺序算法对给定数组中的元素排序。

natsort() 函数实现了“自然排序”，即数字从 1 到 9 的排序方法，字母从 a 到 z 的排序方法，短者优先。数组的索引与单元值保持关联。

如果成功，则该函数返回 TRUE，否则返回 FALSE。

### 语法

natsort(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要进行排序的数组。 |

## 例子

本函数所用的自然排序算法，与通常的计算机字符串排序算法（用于 [sort()](http://www.w3school.com.cn/php/func_array_sort.asp)）的区别，见下面示例：

<?php

$temp\_files = array("temp15.txt","temp10.txt",

"temp1.txt","temp22.txt","temp2.txt");

sort($temp\_files);

echo "Standard sorting: ";

print\_r($temp\_files);

echo "<br />";

natsort($temp\_files);

echo "Natural order: ";

print\_r($temp\_files);

?>

输出：

Standard sorting: Array

(

[0] => temp1.txt

[1] => temp10.txt

[2] => temp15.txt

[3] => temp2.txt

[4] => temp22.txt

)

Natural order: Array

(

[0] => temp1.txt

[3] => temp2.txt

[1] => temp10.txt

[2] => temp15.txt

[4] => temp22.txt

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# next() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

next() 函数把指向当前元素的指针移动到下一个元素的位置，并返回当前元素的值。

如果内部指针已经超过数组的最后一个元素，函数返回 false。

### 语法

next(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

### 说明

next() 和 [current()](http://www.w3school.com.cn/php/func_array_current.asp) 的行为类似，只有一点区别，在返回值之前将内部指针向前移动一位。这意味着它返回的是下一个数组单元的值并将数组指针向前移动了一位。如果移动指针的结果超出了数组单元的末端，则 next() 返回 FALSE。

注意：如果数组包含空的单元，或者单元的值是 0 则该函数碰到这些单元也返回 FALSE。要正确遍历可能含有空单元或者单元值为 0 的数组，请参见 [each() 函数](http://www.w3school.com.cn/php/func_array_each.asp)。

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo current($people) . "<br />";

echo next($people);

?>

输出：

Peter

Joe

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# pos() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

pos() 函数是 [current() 函数](http://www.w3school.com.cn/php/func_array_current.asp) 的别名。它可返回数组中当前元素的值。

### 语法

pos(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo pos($people) . "<br />";

?>

输出：

Peter

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# prev() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

next() 函数把指向当前元素的指针移动到上一个元素的位置，并返回当前元素的值。

如果内部指针已经超过数组的第一个元素之前，函数返回 false。

### 语法

next(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

### 说明

prev() 和 [next()](http://www.w3school.com.cn/php/func_array_next.asp) 的行为类似，不过它将内部指针倒回一位而不是前移一位。

注意：如果数组包含空的单元，或者单元的值是 0 则该函数碰到这些单元也返回 FALSE。要正确遍历可能含有空单元或者单元值为 0 的数组，请参见 [each() 函数](http://www.w3school.com.cn/php/func_array_each.asp)。

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo current($people) . "<br />";

echo next($people) . "<br />";

echo prev($people);

?>

输出：

Peter

Joe

Peter

# [PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)range() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

range() 函数创建并返回一个包含指定范围的元素的数组。

### 语法

range(first,second,step)

|  |  |
| --- | --- |
| **参数** | **描述** |
| first | 必需。规定数组元素的最小值。 |
| second | 必需。规定数组元素的最大值。 |
| step | 可选。规定元素之间的步进制。默认是 1。  注释：该参数是 PHP 5 中加入的。 |

### 说明

该函数创建一个数组，包含从 first 到 second （包含 first 和 second）之间的整数或字符。如果 second 比 first 小，则返回反序的数组。

## 例子 1

<?php

$number = range(0,5);

print\_r ($number);

?>

输出：

Array

(

[0] => 0

[1] => 1

[2] => 2

[3] => 3

[4] => 4

[5] => 5

)

## 例子 2

<?php

$number = range(0,50,10);

print\_r ($number);

?>

输出：

Array

(

[0] => 0

[1] => 10

[2] => 20

[3] => 30

[4] => 40

[5] => 50

)

## 例子 3

<?php

$letter = range("a","d");

print\_r ($letter);

?>

输出：

Array

(

[0] => a

[1] => b

[2] => c

[3] => d

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# reset() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

reset() 函数把数组的内部指针指向第一个元素，并返回这个元素的值。

若失败，则返回 FALSE。

### 语法

reset(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

echo current($people) . "<br />";

echo next($people) . "<br />";

echo reset($people);

?>

输出：

Peter

Joe

Peter

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# rsort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

rsort() 函数对数组的元素按照键值进行逆向排序。与 [arsort()](http://www.w3school.com.cn/php/func_array_arsort.asp) 的功能基本相同。

注释：该函数为 array 中的单元赋予新的键名。这将删除原有的键名而不仅是重新排序。

如果成功则返回 TRUE，否则返回 FALSE。

可选的第二个参数包含另外的排序标志。

### 语法

rsort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。输入的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

rsort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

[0] => Horse

[1] => Dog

[2] => Cat

)

[PHP Array 函](http://www.w3school.com.cn/php/php_ref_array.asp)

# shuffle() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

shuffle() 函数把数组中的元素按随机顺序重新排列。

若成功，则返回 TRUE，否则返回 FALSE。

注释：本函数为数组中的单元赋予新的键名。这将删除原有的键名而不仅是重新排序。

注释：自 PHP 4.2.0 起，不再需要用 srand() 或 mt\_srand() 函数给随机数发生器播种，现已被自动完成。

### 语法

shuffle(array)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要使用的数组。 |

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

shuffle($my\_array);

print\_r($my\_array);

?>

输出：

Array ( [0] => Cat [1] => Horse [2] => Dog )

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# sizeof() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

sizeof() 函数计算数组中的单元数目或对象中的属性个数。

该函数是 [count()](http://www.w3school.com.cn/php/func_array_count.asp) 的别名。

### 语法

sizeof(array,mode)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要计数的数组或对象。 |
| mode | 可选。规定函数的模式。可能的值：   * 0 - 默认。不检测多维数组（数组中的数组）。 * 1 - 检测多维数组。   注释：该参数是 PHP 4.2 中加入的。 |

## 提示和注释

注释：当变量未被设置，或是变量包含一个空的数组，该函数会返回 0。可使用 isset() 变量来测试变量是否被设置。

## 例子

<?php

$people = array("Peter", "Joe", "Glenn", "Cleveland");

$result = sizeof($people);

echo $result;

?>

输出：

4

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# sort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

sort() 函数按升序对给定数组的值排序。

注释：本函数为数组中的单元赋予新的键名。原有的键名将被删除。

如果成功则返回 TRUE，否则返回 FALSE。

### 语法

sort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。输入的数组。 |
| sorttype | 可选。规定如何排列数组的值。可能的值：   * SORT\_REGULAR - 默认。以它们原来的类型进行处理（不改变类型）。 * SORT\_NUMERIC - 把值作为数字来处理 * SORT\_STRING - 把值作为字符串来处理 * SORT\_LOCALE\_STRING - 把值作为字符串来处理，基于本地设置\*。 |

\*：该值是 PHP 4.4.0 和 5.0.2 新加的。在 PHP 6 之前，使用了系统的区域设置，可以用 setlocale() 来改变。自 PHP 6 起，必须用 i18n\_loc\_set\_default() 函数。

## 例子

<?php

$my\_array = array("a" => "Dog", "b" => "Cat", "c" => "Horse");

sort($my\_array);

print\_r($my\_array);

?>

输出：

Array

(

0] => Cat

[1] => Dog

[2] => Horse

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# uasort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

uasort() 函数使用用户自定义的比较函数对数组排序，并保持索引关联（不为元素分配新的键）。

如果成功则返回 TRUE，否则返回 FALSE。

该函数主要用于对那些单元顺序很重要的结合数组进行排序。

### 语法

uasort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要排序的数组。 |
| function | 必需。用户自定义的函数。  函数必须设计为返回 -1, 0, 或 1，并应该接受两个供比较的参数，同时以类似下面这样的方式来工作：   * 如果 a = b, 返回 0 * 如果 a < b, 返回 1 * 如果 a > b, 返回 -1 |

## 例子

<?php

function my\_sort($a, $b)

{

if ($a == $b) return 0;

return ($a > $b) ? -1 : 1;

}

$people = array("Swanson" => "Joe",

"Griffin" => "Peter", "Quagmire" => "Glenn",

"swanson" => "joe", "griffin" => "peter",

"quagmire" => "glenn");

uasort($people, "my\_sort");

print\_r ($people);

?>

输出：

Array

(

[griffin] => peter

[swanson] => joe

[quagmire] => glenn

[Griffin] => Peter

[Swanson] => Joe

[Quagmire] => Glenn

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# uksort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

uksort() 函数使用用户自定义的比较函数按照键名对数组排序，并保持索引关系。

如果成功则返回 TRUE，否则返回 FALSE。

如果要排序的数组需要用一种不寻常的标准进行排序，那么应该使用此函数。

自定义函数应接受两个参数，该参数将被数组中的一对键名填充。比较函数在第一个参数小于，等于，或大于第二个参数时必须分别返回一个小于零，等于零，或大于零的整数。

### 语法

uksort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要排序的数组。 |
| function | 必需。用户自定义的函数。  函数必须设计为返回 -1, 0, 或 1，并应该接受两个供比较的参数，同时以类似下面这样的方式来工作：   * 如果 a = b, 返回 0 * 如果 a > b, 返回 1 * 如果 a < b, 返回 -1 |

## 例子

<?php

function my\_sort($a, $b)

{

if ($a == $b) return 0;

return ($a > $b) ? -1 : 1;

}

$people = array("Swanson" => "Joe",

"Griffin" => "Peter", "Quagmire" => "Glenn",

"swanson" => "joe", "griffin" => "peter",

"quagmire" => "glenn");

uksort($people, "my\_sort");

print\_r ($people);

?>

输出：

Array

(

[swanson] => joe

[quagmire] => glenn

[griffin] => peter

[Swanson] => Joe

[Quagmire] => Glenn

[Griffin] => Peter

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

# usort() 函数

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)

## 定义和用法

usort() 函数使用用户自定义的函数对数组排序。

注释：如果两个元素比较结果相同，则它们在排序后的数组中的顺序未经定义。到 PHP 4.0.6 之前，用户自定义函数将保留这些元素的原有顺序。但是由于在 4.1.0 中引进了新的排序算法，结果将不是这样了，因为对此没有一个有效的解决方案。

注释：本函数为 *array* 中的元素赋予新的键名。这会删除原有的键名。

### 语法

usort(array,sorttype)

|  |  |
| --- | --- |
| **参数** | **描述** |
| array | 必需。规定要排序的数组。 |
| function | 必需。用户自定义的函数。  函数必须设计为返回 -1, 0, 或 1，并应该接受两个供比较的参数，同时以类似下面这样的方式来工作：   * 如果 a = b, 返回 0 * 如果 a > b, 返回 1 * 如果 a < b, 返回 -1 |

## 例子

<?php

function my\_sort($a, $b)

{

if ($a == $b) return 0;

return ($a > $b) ? -1 : 1;

}

$arr = array("Peter", "glenn","Cleveland","peter","cleveland", "Glenn");

usort($arr, "my\_sort");

print\_r ($arr);

?>

输出：

Array

(

[0] => peter

[1] => glenn

[2] => cleveland

[3] => Peter

[4] => Glenn

[5] => Cleveland

)

[PHP Array 函数](http://www.w3school.com.cn/php/php_ref_array.asp)