join »
« htmlspecialchars

- Manual de PHP
- Referencia de funciones
- Procesamiento de texto
- Strings
- Funciones de strings

Change language: Spanish

Submit a Pull Request Report a Bug

implode

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

implode — Une elementos de un array en un string

Descripción ¶

implode(string `$separator`, array `$array`): string

Firma alternativa (no se admite argumentos con nombre):

implode(array `$array`): string

Firma heredada (obsoleta a partir de PHP 7.4.0, eliminada a partir de PHP 8.0.0):

implode(array `$array`, string `$separator`): string

Une los elementos de un array con el string `separator`.

Parámetros ¶

`separator`

Opcional. Por defecto es un string vacío.

`array`

El array de strings a ser usados por implode.

Valores devueltos ¶

Devuelve un string que contiene la representación de todos los elementos del array en el mismo orden, con el string 'glue' entre cada elemento.

Historial de cambios ¶

Versión	Descripción
8.0.0	Pasar el parámetro `separator` después del `array` ya no es compatible.
7.4.0	Pasar el parámetro `separator` después del `array` (es decir, sin utilizar el orden documentado de los parámetros) es obsoleto.

Ejemplos ¶

Ejemplo #1 Ejemplo de implode()

```
<?php

$array = ['lastname', 'email', 'phone'];
var_dump(implode(",", $array)); // string(20) "lastname,email,phone"

// Devuelve un string vacío si se usa un array vacío:
var_dump(implode('hello', [])); // string(0) ""

// El separador es opcional:
var_dump(implode(['a', 'b', 'c'])); // string(3) "abc"

?>
```

Notas ¶

> **Nota**: Esta función es segura binariamente.

Ver también ¶

- explode() - Divide un string en varios string
- preg_split() - Divide un string mediante una expresión regular
- http_build_query() - Generar una cadena de consulta codificada estilo URL

+ add a note

User Contributed Notes 14 notes

up
down
345
houston_roadrunner at yahoo dot com ¶
13 years ago

it should be noted that an array with one or no elements works fine. for example:

```
<?php
    $a1 = array("1","2","3");
    $a2 = array("a");
    $a3 = array();

    echo "a1 is: '".implode("','",$a1)."'<br>";
    echo "a2 is: '".implode("','",$a2)."'<br>";
    echo "a3 is: '".implode("','",$a3)."'<br>";
?>

will produce:
===========
a1 is: '1','2','3'
a2 is: 'a'
a3 is: ''
```

up
down
82
ASchmidt at Anamera dot net ¶
3 years ago

It's not obvious from the samples, if/how associative arrays are handled. The "implode" function acts on the array "values", disregarding any keys:

```
<?php
declare(strict_types=1);

$a = array( 'one','two','three' );
$b = array( '1st' => 'four', 'five', '3rd' => 'six' );

echo implode( ',', $a ),'/', implode( ',', $b );
?>
```

outputs:
one,two,three/four,five,six

up
down
4

biziclop ¶
1 year ago

Sometimes it's necessary to add a string not just between the items, but before or after too, and proper handling of zero items is also needed.
In this case, simply prepending/appending the separator next to implode() is not enough, so I made this little helper function.

```
<?php

function wrap_implode( $array, $before = '', $after = '', $separator = '' ){
  if( ! $array )  return '';
  return $before . implode("{$after}{$separator}{$before}", $array ) . $after;
}

echo wrap_implode(['path','to','file.php'], '/');
// "/path/to/file.php"

$pattern = '#'. wrap_implode([4,2,2], '\d{', '}', '[-.]') .'#';
echo $pattern, "\n";  // #\d{4}[-.]\d{2}[-.]\d{2}#
echo preg_replace( $pattern, '[REDACTED]', 'The UFO appeared between 2012-12-24 and 2013.01.06 every night.');
// 'The UFO appeared between [REDACTED] and [REDACTED] every night.

echo wrap_implode(['line','by','line'], '<b>', '</b>', '<br>  ');
// <b>line</b><br>  <b>by</b><br>  <b>line</b>

echo wrap_implode( ['<a href="">Menu Item 1</a>', '<a href="">Menu Item 2</a>',],
  "<li>", "</li>\n",
  "<li> | </li>\n",
);
/*
<li><a href="">Link1</a></li>
<li> | </li>
<li><a href="">Link2</a></li>
*/

?>
```

up
down
107

omar dot ajoue at kekanto dot com ¶

9 years ago

Can also be used for building tags or complex lists, like the following:

```
<?php

$elements = array('a', 'b', 'c');

echo "<ul><li>" . implode("</li><li>", $elements) . "</li></ul>";

?>
```

This is just an example, you can create a lot more just finding the right glue! ;)

up
down
21

Felix Rauch ¶

6 years ago

It might be worthwhile noting that the array supplied to implode() can contain objects, provided the objects implement the __toString() method.

Example:

```
<?php

class Foo
{
    protected $title;

    public function __construct($title)
    {
        $this->title = $title;
    }

    public function __toString()
    {
        return $this->title;
    }
}

$array = [
    new Foo('foo'),
    new Foo('bar'),
    new Foo('qux')
];

echo implode('; ', $array);
?>
```

will output:

```
foo; bar; qux
```

up
down
41

alexey dot klimko at gmail dot com ¶

11 years ago

If you want to implode an array of booleans, you will get a strange result:

```
<?php
var_dump(implode('',array(true, true, false, false, true)));
```

```
?>

Output:
string(3) "111"

TRUE became "1", FALSE became nothing.
```

up
down
6
Honk der Hase ¶
2 years ago

```
If you want to implode an array as key-value pairs, this method comes in handy.
The third parameter is the symbol to be used between key and value.

<?php
function mapped_implode($glue, $array, $symbol = '=') {
    return implode($glue, array_map(
            function($k, $v) use($symbol) {
                return $k . $symbol . $v;
            },
            array_keys($array),
            array_values($array)
            )
        );
}

$arr = [
    'x'=> 5,
    'y'=> 7,
    'z'=> 99,
    'hello' => 'World',
    7 => 'Foo',
];

echo mapped_implode(', ', $arr, ' is ');

// output: x is 5, y is 7, z is 99, hello is World, 7 is Foo

?>
```

up
down
21
Anonymous ¶
9 years ago

```
It may be worth noting that if you accidentally call implode on a string rather than an array, you
do NOT get your string back, you get NULL:
<?php
var_dump(implode(':', 'xxxxx'));
?>
returns
NULL

This threw me for a little while.
```

up
down
14
masterandujar ¶
10 years ago

Even handier if you use the following:

```
<?php
$id_nums = array(1,6,12,18,24);

$id_nums = implode(", ", $id_nums);

$sqlquery = "Select name,email,phone from usertable where user_id IN ($id_nums)";

// $sqlquery becomes "Select name,email,phone from usertable where user_id IN (1,6,12,18,24)"
?>
```

Be sure to escape/sanitize/use prepared statements if you get the ids from users.

up
down
5

Anonymous ¶
7 years ago

null values are imploded too. You can use array_filter() to sort out null values.

```
<?php
$ar = array("hello", null, "world");
print(implode(',', $ar)); // hello,,world
print(implode(',', array_filter($ar, function($v){ return $v !== null; }))); // hello,world
?>
```

up
down
-7

Rafael Pereira ¶
2 years ago

If you want to use a key inside array:

```
Example:
$arr=array(
array("id" => 1,"name" => "Test1"),
array("id" => 2,"name" => "Test2"),
);

echo implode_key(",",$arr, "name");
OUTPUT: Test1, Test2

function implode_key($glue, $arr, $key){
    $arr2=array();
    foreach($arr as $f){
        if(!isset($f[$key])) continue;
        $arr2[]=$f[$key];
    }
    return implode($glue, $arr2);
}
```

up
down
-36

admin at lanlink dot net dot au ¶
5 years ago

It is possible for an array to have numeric values, as well as string values. Implode will convert all numeric array elements to strings.

```
<?php
```

```
$test=implode(["one",2,3,"four",5.67]);
echo $test;
//outputs: "one23four5.67"
?>
```

up
down
-13
info at ensostudio dot ru ¶
2 years ago

```
<?php
* Join pieces with a string recursively.
*
* @param mixed $glue String between pairs(glue) or an array pair's glue and key/value glue or
$pieces.
* @param iterable $pieces Pieces to implode (optional).
* @return string Joined string
*/
function double_implode($glue, iterable $pieces = null): string
{
    $glue2 = null;
    if ($pieces === null) {
        $pieces = $glue;
        $glue = '';
    } elseif (is_array($glue)) {
        list($glue, $glue2) = $glue;
    }

    $result = [];
    foreach ($pieces as $key => $value) {
        $result[] = $glue2 === null ? $value : $key . $glue2 . $value;
    }
    return implode($glue, $result);
}
?>
Examples:
<?php
$array = ['a' => 1, 'b' => 2];
$str =  implode($array);
$str =  implode(', ', $array);
$str =  implode(['" ', '="'], $array);

$iterator = new ArrayIterator($array);
$str =  implode($iterator);
$str =  implode(', ', $iterator);
$str =  implode(['" ', '="'], $iterator);
?>
```

up
down
-7
info AT sinistercircuits DOT com ¶
1 year ago

```
There is no mention of behavior on a empty array, so I tried it and here's the result:

<?php
$ar = array();
$result = implode(',', $ar);  // Comma arbitrarily applied as the separator
$is_result_empty = empty($result);
?>
```

```
$result:
$is_result_empty: 1

In other words, an empty string is the result.
```

+ add a note

- Other PHP.net sites
- Privacy policy
- View Source