Getting Started
- Introduction
- A simple tutorial

Language Reference
- Basic syntax
- Types
- Variables
- Constants
- Expressions
- Operators
- Control Structures
- Functions
- Classes and Objects
- Namespaces
- Enumerations
- Errors
- Exceptions
- Fibers
- Generators
- Attributes
- References Explained
- Predefined Variables
- Predefined Exceptions
- Predefined Interfaces and Classes
- Predefined Attributes
- Context options and parameters
- Supported Protocols and Wrappers

Security
- Introduction
- General considerations
- Installed as CGI binary
- Installed as an Apache module
- Session Security
- Filesystem Security
- Database Security
- Error Reporting
- User Submitted Data
- Hiding PHP
- Keeping Current

Features
- HTTP authentication with PHP
- Cookies
- Sessions
- Dealing with XForms
- Handling file uploads
- Using remote files
- Connection handling
- Persistent Database Connections
- Command line usage

Garbage Collection
DTrace Dynamic Tracing

Function Reference
Affecting PHP's Behaviour
Audio Formats Manipulation
Authentication Services
Command Line Specific Extensions
Compression and Archive Extensions
Cryptography Extensions
Database Extensions
Date and Time Related Extensions
File System Related Extensions
Human Language and Character Encoding Support
Image Processing and Generation
Mail Related Extensions
Mathematical Extensions
Non-Text MIME Output
Process Control Extensions
Other Basic Extensions
Other Services
Search Engine Extensions
Server Specific Extensions
Session Extensions
Text Processing
Variable and Type Related Extensions
Web Services
Windows Only Extensions
XML Manipulation
GUI Extensions

Keyboard Shortcuts

?	This help
j	Next menu item
k	Previous menu item
g p	Previous man page
g n	Next man page
G	Scroll to bottom
g g	Scroll to top
g h	Goto homepage
g s	Goto search (current page)
/	Focus search box

switch »
« break

Change language: Russian

continue

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

`continue` используется внутри циклических структур для пропуска оставшейся части текущей итерации цикла и, при соблюдении условий, начала следующей итерации.

> **Замечание**: В PHP оператор switch считается циклическим и внутри него может использоваться `continue`. Если `continue` не передано аргументов, то он ведёт себя аналогично `break`, но выдаёт предупреждение о возможной ошибке. Если `switch` расположен внутри цикла, `continue 2` продолжит выполнение внешнего цикла со следующей итерации.

`continue` принимает необязательный числовой аргумент, который указывает на скольких уровнях вложенных циклов будет пропущена оставшаяся часть итерации. Значением по умолчанию является `1`, при которой пропускается оставшаяся часть текущего цикла.

```
<?php
$arr = ['ноль', 'один', 'два', 'три', 'четыре', 'пять', 'шесть'];
foreach ($arr as $key => $value) {
if (0 === ($key % 2)) { // пропуск чётных чисел
continue;
}
echo $value . "\n";
}
?>
```

Результат выполнения приведённых примеров:

```
один
три
пять
```

```
<?php
$i = 0;
while ($i++ < 5) {
echo "Внешний\n";
while (1) {
echo "Средний\n";
while (1) {
echo "Внутренний\n";
continue 3;
}
echo "Это никогда не выведется.\n";
}
echo "Это также не выведется.\n";
}
?>
```

Результат выполнения приведённых примеров:

```
Внешний
Средний
Внутренний
Внешний
Средний
Внутренний
Внешний
Средний
Внутренний
Внешний
Средний
Внутренний
Внешний
Средний
Внутренний
```

Пропуск точки запятой после `continue` может привести к путанице. Пример как не надо делать.

```
<?php
for ($i = 0; $i < 5; ++$i) {
if ($i == 2)
continue
print "$i\n";
}
?>
```

Ожидается, что результат будет такой:

```
0
1
3
4
```

Изменения, касающиеся оператора `continue`

Версия	Описание
7.3.0	`continue` внутри `switch`, использующееся как замена `break` для `switch` будет вызывать ошибку уровня **`E_WARNING`**.

+add a note

User Contributed Notes 7 notes

up
down
138
jaimthorn at yahoo dot com ¶
13 years ago

```
The remark "in PHP the switch statement is considered a looping structure for the purposes of continue" near the top of this page threw me off, so I experimented a little using the following code to figure out what the exact semantics of continue inside a switch is:

<?php

for( $i = 0; $i < 3; ++ $i )
{
echo ' [', $i, '] ';
switch( $i )
{
case 0: echo 'zero'; break;
case 1: echo 'one' ; XXXX;
case 2: echo 'two' ; break;
}
echo ' <' , $i, '> ';
}

?>

For XXXX I filled in

- continue 1
- continue 2
- break 1
- break 2

and observed the different results. This made me come up with the following one-liner that describes the difference between break and continue:

continue resumes execution just before the closing curly bracket ( } ), and break resumes execution just after the closing curly bracket.

Corollary: since a switch is not (really) a looping structure, resuming execution just before a switch's closing curly
```

bracket has the same effect as using a break statement. In the case of (for, while, do-while) loops, resuming execution just prior their closing curly brackets means that a new iteration is started --which is of course very unlike the behavior of a break statement.

In the one-liner above I ignored the existence of parameters to break/continue, but the one-liner is also valid when parameters are supplied.

up
down
43

Nikolay Ermolenko ¶

14 years ago

Using continue and break:

```
<?php
$stack = array('first', 'second', 'third', 'fourth', 'fifth');

foreach($stack AS $v){
if($v == 'second')continue;
if($v == 'fourth')break;
echo $v.'<br>';
}
/*

first
third

*/

$stack2 = array('one'=>'first', 'two'=>'second', 'three'=>'third', 'four'=>'fourth', 'five'=>'fifth');
foreach($stack2 AS $k=>$v){
if($v == 'second')continue;
if($k == 'three')continue;
if($v == 'fifth')break;
echo $k.' ::: '.$v.'<br>';
}
/*

one ::: first
four ::: fourth

*/

?>
```

up
down
21

Koen ¶

11 years ago

If you use a incrementing value in your loop, be sure to increment it before calling continue; or you might get an infinite loop.

up
down
18

rjsteinert.com ¶

12 years ago

The most basic example that print "13", skipping over 2.

```
<?php
$arr = array(1, 2, 3);
foreach($arr as $number) {
if($number == 2) {
continue;
}
print $number;
}
?>
```
up
down
11
www.derosetechnologies.com ¶
19 years ago
In the same way that one can append a number to the end of a break statement to indicate the "loop" level upon which one wishes to 'break' , one can append a number to the end of a 'continue' statement to acheive the same goal. Here's a quick example:

```
<?
for ($i = 0;$i<3;$i++) {
echo "Start Of I loop\n";
for ($j=0;;$j++) {

if ($j >= 2) continue 2; // This "continue" applies to the "$i" loop
echo "I : $i J : $j"."\n";
}
echo "End\n";
}
?>
```

The output here is:

```
Start Of I loop
I : 0 J : 0
I : 0 J : 1
Start Of I loop
I : 1 J : 0
I : 1 J : 1
Start Of I loop
I : 2 J : 0
I : 2 J : 1
```

For more information, see the php manual's entry for the 'break' statement.
up
down
2
tufan dot oezduman at gmail dot com ¶
17 years ago
a possible explanation for the behavior of continue in included scripts mentioned by greg and dedlfix above may be the following line of the "return" documentation: "If the current script file was include()ed or require()ed, then control is passed back to the calling file."
The example of greg produces an error since page2.php does not contain any loop-operations.

So the only way to give the control back to the loop-operation in page1.php would be a return.
up
down
0
Geekman ¶
16 years ago
For clarification, here are some examples of continue used in a while/do-while loop, showing that it has no effect on the conditional evaluation element.

```
<?php
// Outputs "1 ".
$i = 0;
while ($i == 0) {
$i++;
echo "$i ";
if ($i == 1) continue;
}

// Outputs "1 2 ".
$i = 0;
do {
$i++;
echo "$i ";
if ($i == 2) continue;
} while ($i == 1);
?>

Both code snippets would behave exactly the same without continue.
```

+add a note

- Управляющие конструкции
  - Введение
  - if
  - else
  - elseif/else if
  - Альтернативный синтаксис управляющих структур
  - while
  - do-while
  - for
  - foreach
  - break
  - continue
  - switch
  - match
  - declare
  - return
  - require
  - include
  - require_once
  - include_once
  - goto