



- [Downloads](#)
- [Documentation](#)
- [Get Involved](#)
- [Help](#)
- 

[Dutch PHP Conference 2024](#)

[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

[elseif/else if »](#)

[« if](#)

- [Руководство по PHP](#)
- [Справочник языка](#)
- [Управляющие конструкции](#)

Change language: Russian

else

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

Часто необходимо выполнить одно выражение, если условие верно, и другое — если неверно. Вот для чего нужна конструкция `else`. Конструкция `else` расширяет конструкцию `if`, чтобы выполнять другое выражение тогда, когда условие внутри `if` оценивается как **false**. Например, следующий код выведет «а больше b», если значение переменной `$a` больше, чем значение переменной `$b`, иначе — «а НЕ больше b»:

```
<?php

if ($a > $b) {
    echo "а больше b";
} else {
    echo "а НЕ больше b";
}

?>
```

Выражение `else` выполняется, только если условие `if` вычисляется как **false**, а если были условия `elseif` — то только если они тоже вычисляются как **false**. [О конструкции elseif](#).

Замечание: Болтающийся else

В случае вложенных конструкций `if-else`, конструкция `else` связывается с близлежащей конструкцией `if`.

```
<?php

$a = false;
$b = true;
if ($a)
if ($b)
    echo "b";
else
    echo "c";

?>
```

Независимо от расстановки отступов, которые не влияют на PHP-код, конструкция `else` связана с конструкцией `if ($b)`, поэтому пример ничего не выведет. Код с такой расстановкой отступов будет работать, но лучше избегать такого кода и использовать фигурные скобки, чтобы устранить неоднозначности.

[+add a note](#)

User Contributed Notes 2 notes

[up](#)
[down](#)

20
[dormeydo at gmail dot com ¶](#)
15 years ago

An alternative and very useful syntax is the following one:

```
statement ? execute if true : execute if false
```

This is very usefull for dynamic outout inside strings, for example:

```
print('$a is ' . ($a > $b ? 'bigger than' : ($a == $b ? 'equal to' : 'smaler than' )) . ' $b');
```

This will print "\$a is smaler than \$b" is \$b is bigger than \$a, "\$a is bigger than \$b" if \$a si bigger and "\$a is equal to \$b" if they are same.

[up](#)

[down](#)

11

[Caliban Darklock](#)

19 years ago

If you're coming from another language that does not have the "elseif" construct (e.g. C++), it's important to recognise that "else if" is a nested language construct and "elseif" is a linear language construct; they may be compared in performance to a recursive loop as opposed to an iterative loop.

```
<?php
$limit=1000;
for($idx=0;$idx<$limit;$idx++)
{ $list[]="if(false) echo \"$idx;\n\"; else"; }
$list[]=" echo \"$idx\n\";";
$space=implode(" ", $list);| // if ... else if ... else
$nospace=implode("", $list); // if ... elseif ... else
$start=array_sum(explode(" ", microtime()));
eval($space);
$end=array_sum(explode(" ", microtime()));
echo $end-$start . " seconds\n";
$start=array_sum(explode(" ", microtime()));
eval($nospace);
$end=array_sum(explode(" ", microtime()));
echo $end-$start . " seconds\n";
?>
```

This test should show that "elseif" executes in roughly two-thirds the time of "else if". (Increasing \$limit will also eventually cause a parser stack overflow error, but the level where this happens is ridiculous in real world terms. Nobody normally nests if() blocks to more than a thousand levels unless they're trying to break things, which is a whole different problem.)

There is still a need for "else if", as you may have additional code to be executed unconditionally at some rung of the ladder; an "else if" construction allows this unconditional code to be elegantly inserted before or after the entire rest of the process. Consider the following elseif() ladder:

```
<?php
if($a) { conditional1(); }
elseif($b) { conditional2(); }
elseif($c) { conditional3(); }
elseif($d) { conditional4(); }
elseif($e) { conditional5(); }
elseif($f) { conditional6(); }
elseif($g) { conditional7(); }
elseif($h) { conditional8(); }
else { conditional9(); }
?>
```

To insert unconditional preprocessing code for \$e onward, one need only split the "elseif":

```
<?php
if($a) { conditional1(); }
elseif($b) { conditional2(); }
elseif($c) { conditional3(); }
elseif($d) { conditional4(); }
else {
...unconditional();
...if($e) { conditional5(); }
...elseif($f) { conditional6(); }
...elseif($g) { conditional7(); }
...elseif($h) { conditional8(); }
...else { conditional9(); }
}
?>
```

The alternative is to duplicate the unconditional code throughout the construct.

[+ 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](#)
- [Copyright © 2001-2024 The PHP Group](#)
- [My PHP.net](#)
- [Contact](#)
- [Other PHP.net sites](#)
- [Privacy policy](#)

