Search

elseif/else if »
« if

- Руководство по PHP
- Справочник языка
- Управляющие конструкции

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 «a больше b», если значение переменной *$a* больше, чем значение переменной *$b*, иначе — «a НЕ больше b»:

```
<?php

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

Ths 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.