ceil

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

ceil — Redondear fracciones hacia arriba

Descripción ¶

ceil(float `$value`): float

Devuelve el siguiente valor entero mayor redondeando hacia arriba `value` si fuera necesario.

Parámetros ¶

`value`

El valor a redondear

Valores devueltos ¶

`value` redondeado al siguiente entero más alto. El valor de retorno de **ceil()** sigue siendo de tipo float ya que el rango de valores de float es usualmente mayor que el del tipo integer.

Ejemplos ¶

Ejemplo #1 Ejemplo de ceil()

```php
<?php
echo ceil(4.3); // 5
echo ceil(9.999); // 10
echo ceil(-3.14); // -3
?>
```

Ver también ¶

- floor() - Redondear fracciones hacia abajo
- round() - Redondea un float

+ add a note

User Contributed Notes 21 notes

up

down
85
Scott Weaver / scottmweaver * gmail ¶
14 years ago

I needed this and couldn't find it so I thought someone else wouldn't have to look through a bunch of Google results-

```
<?php

// duplicates m$ excel's ceiling function
if( !function_exists('ceiling') )
{
    function ceiling($number, $significance = 1)
    {
        return ( is_numeric($number) && is_numeric($significance) ) ?
(ceil($number/$significance)*$significance) : false;
    }
}

echo ceiling(0, 1000);     // 0
echo ceiling(1, 1);        // 1000
echo ceiling(1001, 1000);  // 2000
echo ceiling(1.27, 0.05);  // 1.30

?>
```

up
down
37
eep2004 at ukr dot net ¶
4 years ago

Caution!

```
<?php
$value = 77.4;
echo ceil($value * 100) / 100;         // 77.41 - WRONG!
echo ceil(round($value * 100)) / 100;  // 77.4 - OK!
```

up
down
23
steve_phpnet // nanovox \\ com ¶
18 years ago

I couldn't find any functions to do what ceiling does while still leaving I specified number of decimal places, so I wrote a couple functions myself. round_up is like ceil but allows you to specify a number of decimal places. round_out does the same, but rounds away from zero.

```
<?php
// round_up:
// rounds up a float to a specified number of decimal places
// (basically acts like ceil() but allows for decimal places)
function round_up ($value, $places=0) {
  if ($places < 0) { $places = 0; }
  $mult = pow(10, $places);
  return ceil($value * $mult) / $mult;
}

// round_out:
// rounds a float away from zero to a specified number of decimal places
function round_out ($value, $places=0) {
  if ($places < 0) { $places = 0; }
  $mult = pow(10, $places);
  return ($value >= 0 ? ceil($value * $mult):floor($value * $mult)) / $mult;
}

echo round_up (56.77001, 2); // displays 56.78
echo round_up (-0.453001, 4); // displays -0.453
echo round_out (56.77001, 2); // displays 56.78
echo round_out (-0.453001, 4); // displays -0.4531
?>
```

up
down
11
***oktam* ¶**
11 years ago

```
Actual behaviour:
echo ceil(-0.1); //result "-0" but i expect "0"

Workaround:
echo ceil(-0.1)+0; //result "0"
```

up
down
6
***aaron at mind-design dot co dot uk* ¶**
18 years ago

```
Or for the terniary fans:

<?php

function roundaway($num) {
   return(($num > 0) ? ceil($num) : floor($num));
}

?>

Slightly pointless, but there you have it, in one line only..
```

up
down
5
***frozenfire at php dot net* ¶**
11 years ago

Please see http://www.php.net/manual/en/language.types.float.php for information regarding floating point precision issues.

up
down
3
***Gemini_13 at torba dot su* ¶**
8 years ago

```
$test = (1 - 0.7) * 10;

$test_1 = ceil($test);
$test_2 = ceil($test * 10);

var_dump($test_1);     // float 4
var_dump($test_2);     // float 31
```

up
down
2
***eep2004 at ukr dot net* ¶**

4 years ago

```
Caution!
<?php
$value = 77.4;
echo ceil($value * 100) / 100;             // 77.41 - WRONG!
echo ceil((string)($value * 100)) / 100;  // 77.4 - OK!
```

up
down
2
themanwe at yahoo dot com ¶
15 years ago

```
float ceil

function fCeil($val,$pressision=2){
     $p = pow(10,$pressision);
    $val = $val*$p;
    $val = ceil($val);
  return $val /$p;
}
```

up
down
1
Bas Vijfwinkel ¶
7 years ago

```
Note that 'ceil' can show unexpected behaviour when using float values due to inaccuracies in the
float system and PHP tendency to freely limiting the number of digits it displays to the user.
It might like it is rounding up values that end in 0 (zero) but actually PHP is not showing that
the value is not exactly the value it shows.

<?PHP
echo('Displaying 13915.0000000000018190 : '. 13915.0000000000018190);
echo('<br>');
echo('Result ceil(13915.0000000000018190) :'.ceil(13915.0000000000018190));
echo('<br>');
?>
For example 13915 cannot be represented as exactly 13915 but becomes
13916.0000000000018190 and 'ceil' will therefore round it up to 13916.

<?PHP
$value = 12650 * 1.1;
echo('Expected behaviour : ceil(12650 * 1.1) = ceil(13915) = 13915');
echo('<br>');
echo('Using only ceil :'.ceil($value));
echo('<br>');
echo('Showing decimals : '.number_format($value - floor($value), 16));
echo('<br>');
echo('Using ceil + round : '.ceil(round($value,4)));
?>

So if 'ceil' looks like it is not working correctly and rounding up figures that end in 0 (zero)
then this might be the cause of this behaviour.
Adding a simple 'round' before the applying the 'ceil' solves this problem.
```

up
down
3
Lexand ¶
10 years ago

```
$k = 0.14 * 100;
echo ceil($k); // results 15

solution is in converting float number to string

Example 1.
echo ceil ("{$k}"); // results 14

Example 2.
$totalSum1 = 102.1568;
$k = $totalSum1 / 100;
echo ceil ("{$k}"); // results 102.16

Example 3.
$totalSum2 = 102.15;
$k = $totalSum1 / 100;
echo ceil ("{$k}"); // results 102.15

useful for 'ceil' with precision capability
```

up
down
2
roger_dupere at hotmail dot com ¶
19 years ago

```
Here is a navbar using the ceil function.

<?php
function navbar($num_rows,$page,$link) {
   $nbrlink = 10; /* Number of link to display per page */
   $page = (int) $page; /* Page now displayed */
   $num_rows = (int) $num_rows;

   if( $num_rows > 0 ) {
     $total_page = ceil( $num_rows / $nbrlink );

     for( $i=1;$i<$total_page+1;$i++ ) {
       if( $i == $page ) {
         $ret .= " <b>$i</b> ";
       } else {
         if( strstr( $link,"?" ) ) {
           $ret .= " <a href=\"$link&page=$i\">$i</a> ";
         } else {
           $ret .= " <a href=\"$link?page=$i\">$i</a> ";
         }
       }
     }

     return $ret;
   }
}
  /* Let say that $num_rows content the numbre of rows of your sql query */
  $navbar = navbar( $num_rows, $page, "listmovie.php?id=$id" );

  if( $navbar != null || $navbar != "" ) {
    print( "<p><div align=\"center\">$navbar</div></p>" );
  }
?>
```

up

down
1
rainfalling at yahoo dot com ¶
18 years ago

```
IceKarma said: "If you want, say, 2.6 to round to 3, and -2.6 to round to -3, you want round(),
which rounds away from zero."

That's not always true. round() doesn't work that way, like zomis2k said it just rounds up _or_
down to the nearest non-decimal number. However this should work.

<?php

function roundaway($num) {
    if ($num > 0)
      return ceil($num);
    elseif ($num < 0)
      return floor($num);
    elseif ($num == 0)
      return 0;
}

?>
```

up
down
1
php is the best ¶
10 years ago

```
Ceil for decimal numbers with precision:

function ceil_dec($number,$precision,$separator)
{
    $numberpart=explode($separator,$number);
$numberpart[1]=substr_replace($numberpart[1],$separator,$precision,0);
    if($numberpart[0]>=0)
    {$numberpart[1]=ceil($numberpart[1]);}
    else
    {$numberpart[1]=floor($numberpart[1]);}

    $ceil_number= array($numberpart[0],$numberpart[1]);
    return implode($separator,$ceil_number);
}

echo ceil_dec(1.125,2,"."); //1.13
echo ceil_dec(-1.3436,3,"."); //-1.343
echo ceil_dec(102938.1,4,"."); //102938.1
```

up
down
0
cedricaubert at gmail dot com ¶
7 years ago

```
Here is a more accurate way to round on superior decimal.

function round_sup($nb, $precision) {
        $p = pow(10, $precision);
        return ceil(round($nb * $p, 1)) / $p;
}

$k = 10 * 4.98;                    // k = 49.8
```

```
echo ceil($k * 10) / 10;    // 49.9   !!
echo round_sup($k, 1);   // 49.8
```

up
down
0
eg at pensio dot com ¶
9 years ago

```
Remember that floating point precision means behavior can be "correct" - though not what you
expect:

php > echo 100 * 1 * 0.07;
7
php > echo ceil(100 * 1 * 0.07);
8
```

up
down
0
alesegdia at gmail dot com ¶
9 years ago

```
Some people asking on rounding -1.5 to -2 and 1.5 to 2, the way is this:

<?=round($var, 0, PHP_ROUND_HALF_UP)?>

See round() doc for more information on this.
```

up
down
0
AndrewS ¶
11 years ago

```
The code below rounds a value up to a nearest multiple, away from zero.  The multiple does not
have to be a integer.  So you could round, say, to the nearest 25.4, allowing you to round
measurements in mm to the nearest inch longer.

<?php
// $x is the variable
// $c is the base multiple to round to, away from zero
$result =  ( ($y = $x/$c) == ($y = (int)$y) ) ? $x : ( $x>=0 ?++$y:--$y)*$c ;
?>

I originally developed this as an example of write-only code: to make the point that being
cleverly terse might save clock ticks but wastes more in programmer time generating un-
maintainable code.

The inline code above nests one conditional statement inside another.  The value of y changes
twice within the same line (three times, if you count the pre-increment).  The value of each
assignment is used to determine branching within the conditional statement.

How it works can more easily be seen from the expansion below:

<?php
function myCeilingLong($x,$c)
{
    // $x is variable
    // $c is ceiling multiple
    $a = $x/$c ;
    $b = (int)$a ;
    if ($a == $b)
        return $x ;  // x is already a multiple of c;
    else
    {
        if ($x>=0)
            return ($b+1)*$c ;  // return ((int)(x/c)+1 ) * c
        else
            return ($b-1)*$c ;  // return ((int)(x/c)-1 ) * c
    }
}
?>

<?php
function myCeilingShort($x,$c)
{
    return ( ($y = $x/$c) == ($y = (int)$y) ) ? $x : ( $x>=0 ?++$y:--$y)*$c ;
}
?>
```

Comparing the versions for speed: the in-line version is about three times faster than myCeilingLong() - but this is almost entirely down to function call overhead.

Putting the in-line code inside the function: the difference in execution speed between myCeilingLong() and myCeilingShort() is around 1.5%.

ceil() is still around 25% faster than the in-line statement so if you are a speed hound your efforts might be better devoted to compiling your own library ...

up
down
-1
sebastien dot thevenaz at gmail dot com ¶
10 years ago
Here is another way to use ceil for decimal numbers with precision:

```
<?php
    function ceil_dec($number, $precision)
    {
        $coefficient = pow(10,$precision);
        return ceil($number*$coefficient)/$coefficient;
    }
?>
```

up
down
-6
Anonymous ¶
7 years ago
Just a quick note to be careful of, if you use the "round_up" code suggested by steve, i must warn you it isn't completely fool proof.

I have the following maths which is evaluated incorrectly

37.2000 - 6.2000 = 31.01

```
echo(round_up(37.2000 - 6.2000,2));
```

This returns the incorrect result, 31.01 which as any junior in mathematics knows...IS WRONG!

use with caution.
up
down

-12
josoort at home dot nl ¶
8 years ago

```
You can simply make a ceil with decimals function by using the round() function and add 0.05 to
the value if the value must have 1 decimal accuracy, or 0.005 for 2 decimals accuracy, or 0.0005
for 3 decimals etc.

Example :

function ceilDec($input, $decimals)
{
    $subtract = 5 / pow(10, $decimals + 1);
    $output = round($input + $subtract, $decimals);
    return $output;
}
```

+ add a note

  - sqrt
  - tan
  - tanh

- Copyright © 2001-2023 The PHP Group
- My PHP.net
- Contact
- Other PHP.net sites
- Privacy policy