Operaciones aritméticas

Números

Los números decimales se escriben con punto (.), no con coma (,).

Volver al principio de la página

Operadores aritméticos

Los operadores aritméticos básicos son los siguientes:

Ejemplo	Nombre	Resultado
-$a	Negación	El opuesto de $a.
$a + $b	Suma	Suma de $a y $b.
$a - $b	Resta	Diferencia entre $a y $b.
$a * $b	Multiplicación	Producto de $a y $b.
$a / $b	División	Cociente de $a y $b.
$a % $b	Módulo	Resto de $a dividido por $b. Cuidado: Los números se convierten a enteros antes de efectuar el cálculo. Por ejemplo, 5 % 2.5 da como resultado 1 y no 0 porque calcula el resto de 5 entre 2, no de 5 entre 2.5.

Hay que tener en cuenta que en PHP un entero no puede ser arbitrariamente grande. A partir de cierto valor, que depende del sistema operativo, PHP convierte automáticamente los enteros en float, perdiéndose precisión. En un sistema de 32 bits, el valor máximo es 2147483647 (2^{31}-1).

Si se necesita trabajar con enteros mayores, es necesario utilizar las funciones bcmath.

Volver al principio de la página

Operadores de incremento y decremento

Ejemplo	Nombre	Efecto
++$a	Pre-incremento	Incrementa $a en uno, y luego devuelve $a.
$a++	Post-incremento	Devuelve $a, y luego incrementa $a en uno.
--$a	Pre-decremento	Decrementa $a en uno, luego devuelve $a.
$a--	Post-decremento	Devuelve $a, luego decrementa $a en uno.

La diferencia entre el pre-incremento y el post-incremento es que en el primer caso primero se incrementa la variable y después se utiliza y en el segundo primero se utiliza y después se incrementa.

```
<?php
$valor = 9;
print "<p>" . ++$valor . "</p>\n";
?>
```

```
<p>10</p>
```

```
<?php
$valor = 9;
print "<p>" . $valor++ . "</p>\n";
?>
```

```
<p>9</p>
```

El operador de incremento funciona también con caracteres, teniendo en cuenta que al incrementar el carácter 'z' se obtiene la cadena 'aa'. El operador de decremento no funciona con caracteres.

```
<?php
$valor = "b";
print "<p>" . ++$valor . "</p>\n";
?>
```

```
<p>c</p>
```

```
<?php
$valor = "z";
print "<p>" . ++$valor . "</p>\n";
?>
```

```
<p>aa</p>
```

```
<?php
$valor = "a9z";
print "<p>" . ++$valor . "</p>\n";
?>
```

```
<p>b0a</p>
```

Volver al principio de la página

Resto de una división

El operador % calcula el resto de una división entera

```
<?php
$resto = 17 % 3;
print "<p>El resto de 17 dividido entre 3 es " .
$resto . "</p>\n";
?>
```

```
<p>El resto de 17 dividido entre
3 es 2</p>
```

La función fmod calcula el resto de una división con números decimales

```
<?php
print "<p>El resto de 17 dividido entre 3.1 es " .
fmod(17, 3.1) . "</p>\n";
?>
```

```
<p>El resto de 17 dividido entre
3.1 es 1.5</p>
```

Hay que tener en cuenta que en PHP un entero no puede ser arbitrariamente grande. A partir de cierto valor, que depende del sistema operativo, PHP convierte automáticamente los enteros en float, perdiéndose precisión. En un sistema de 32 bits, el valor máximo es 2147483647 (2^{31}-1).

Si se necesita trabajar con enteros mayores, es necesario utilizar las funciones bcmath.

Volver al principio de la página

Paréntesis

Los operadores artiméticos tienen el mismo orden de precedencia que en Matemáticas. Concretamente, el orden de precedencia de los operadores comentados anteriormente es, de mayor a menor, el siguiente (los operadores indicados en el mismo grupo se efectúan en el orden en que aparecen en la expresión):

- ++ (incremento) -- (decremento)
- * (multiplicación) / (división) % (resto)

- + (suma) - (resta)

Los paréntesis permiten agrupar operaciones de manera que las operaciones entre paréntesis se realicen antes que las operaciones fuera de los paréntesis. Se aconseja no utilizar paréntesis cuando las operaciones den el mismo resultado con o sin paréntesis.

Volver al principio de la página

Operadores de asignación

Los operadores de asignación permiten simplificar algunas expresiones de asignación:

Ejemplo	Nombre	Equivale a
$a += $b	Suma	$a = $a + $b
$a -= $b	Resta	$a = $a - $b
$a *= $b	Multiplicación	$a = $a * $b
$a /= $b	División	$a = $a / $b
$a %= $b	Módulo	$a = $a % $b

Volver al principio de la página

Redondear un número

La función `round(x)` redondea el número x al entero más próximo.

```
<?php
print "<p>2.6 se redondea con round a " . round(2.6) .
"</p>\n";
print "<p>2.3 se redondea con round a " . round(2.3) .
"</p>\n";
?>
```

```
<p>2.3 se redondea con round a
2</p>
<p>2.6 se redondea con round a a
3</p>
```

La función `round()` se puede utilizar para verificar si un número es un número entero (positivo o negativo), comprobando si un número coincide con su valor redondeado.

```
<?php
$numero = 4.3;
if ($numero == round($numero)) {
    print "<p>$numero es un número entero</p>\n";
} else {
    print "<p>$numero no es un número entero</p>\n";
}

$numero = -6;
if ($numero == round($numero)) {
    print "<p>$numero es un número entero</p>\n";
} else {
    print "<p>$numero no es un número entero</p>\n";
}
?>
```

```
<p>4.3 no es un número entero</p>
<p>-6 es un número entero</p>
```

La función `round(x,n)` redondea x con n decimales (si n es negativo redondea a decenas, centenas, etc.).

```
<?php
print "<p>2.6574 se redondea con round con dos
decimales a " . round(2.6574, 2) . "</p>\n";
print "<p>3141592 redondeado con round con centenas
es " . round(3141592, -2) . "</p>\n";
?>
```

```
<p>2.6574 se redondea con round
con dos decimales a 2.66</p>
<p>3141592 redondeado con round
con centenas es 3141600</p>
```

La función `floor(x)` redondea el número x al entero inferior (es decir, devuelve la parte entera).

```
<?php
print "<p>2.6 se redondea con floor a " . floor(2.6) .
```

```
<p>2.6 se redondea con floor a
2</p>
```

```
"</p>\n";
print "<p>2.3 se redondea con floor a " . floor(2.3) .
"</p>\n";
print "<p>-2.6 se redondea con floor a " . floor(-2.6)
. "</p>\n";
print "<p>-2.3 se redondea con floor a " . floor(-2.3)
. "</p>\n";
?>
```

```
<p>2.3 se redondea con floor a
2</p>
<p>-2.6 se redondea con floor a
-3</p>
<p>-2.3 se redondea con floor a
-3</p>
```

La función ceil(x) redondea el número x al entero superior.

```
<?php
print "<p>2.6 se redondea con ceil a " . ceil(2.6) .
"</p>\n";
print "<p>2.3 se redondea con ceil a " . ceil(2.3) .
"</p>\n";
print "<p>-2.6 se redondea con ceil a " . ceil(-2.6) .
"</p>\n";
print "<p>-2.3 se redondea con ceil a " . ceil(-2.3) .
"</p>\n";
?>
```

```
<p>2.6 se redondea con ceil a
3</p>
<p>2.3 se redondea con ceil a
3</p>
<p>-2.6 se redondea con ceil a
-2</p>
<p>-2.3 se redondea con ceil a
-2</p>
```

Volver al principio de la página

Potencias

La función pow(x, y) calcula x elevado a y.

```
<?php
print "<p>2<sup>3</sup> = " . pow(2, 3) . "</p>\n";
?>
```

```
<p>2 = 8</p>
```

Volver al principio de la página

Máximo y mínimo

Las funciones max() y min() devuelven el máximo y el mínimo, respectivamente, de una lista o matriz de valores..

```
<?php
print "<p>El máximo es " . max(20, 40, 25.1, 14.7) . "</p>\n";
?>
```

```
<p>El máximo es 40</p>
```

```
<?php
print "<p>El mínimo es " . min(20, 40, 25.1, 14.7) . "</p>\n";
?>
```

```
<p>El mínimo es 14.7</p>
```

```
<?php
$datos = array(20, 40, 25.1, 14.7);
print "<p>El mínimo es " . min($datos) . "</p>\n";
?>
```

```
<p>El mínimo es 14.7</p>
```

Volver al principio de la página

Formatear un número

Para escribir un número con los símbolos de separación de decimales y de miles españoles, es decir, una coma (,) para separar la parte entera de la decimal y un punto (.) para separar las cifras de la parte entera en grupos de tres, se puede utilizar la función number_format().

```
<?php
print "<p>" . number_format(1300, 5) . "</p>\n";
?>
```

```
<p>1,300.00000</p>
```

```
<?php
print "<p>" . number_format(123456.789, 2) . "</p>\n";
?>
```

```
<p>123,456.79</p>
```

```
<?php
print "<p>" . number_format(123456789123, 0, ',', '.') .
"</p>\n";
?>
```

```
<p>123.456.789.123</p>
```

```
<?php
print "<p>" . number_format(123456789123456.789, 2, ',',
'.') . "</p>\n";
?>
```

```
<p>123.456.789.123.456,78</p>
```

La función requiere dos o cuatro argumentos:

- el primer argumento es el número a formatear.
- el segundo argumento es el número de decimales a mostrar (el número se redondea o trunca dependiendo de la longitud del número, compárese el segundo y el cuarto ejemplo de los ejemplos anteriores).
- el tercer argumento es el carácter a utilizar como separador de la parte entera de la decimal.
- el cuarto argumento es el carácter a utilizar como separador de miles.

La función devuelve el número formateado. Si sólo se utilizan dos argumentos, se utiliza el punto como separador de parte entera y decimal y la coma como separador de miles (notación inglesa).

Esta página forma parte del curso "Páginas web con PHP" disponible en http://www.mclibre.org
Autor: Bartolomé Sintes Marco
Última modificación de esta página: 24 de octubre de 2013

Esta obra está bajo una licencia de Creative Commons Reconocimiento-CompartirIgual 3.0 España.