Opérateurs PHP

Dans ce chapitre, vous apprendrez à manipuler ou à effectuer des opérations sur des variables et des valeurs à l'aide d'opérateurs en PHP.

Qu'est-ce que les opérateurs en PHP

Les opérateurs sont des symboles qui indiquent au processeur PHP d'effectuer certaines actions. Par exemple, le symbole d'addition (+) est un opérateur qui indique à PHP d'ajouter deux variables ou valeurs, tandis que le symbole supérieur à (>) est un opérateur qui indique à PHP de comparer deux valeurs.

Les slides suivants décrivent les différents opérateurs utilisés en PHP.

Opérateurs arithmétiques PHP

Les opérateurs arithmétiques sont utilisés pour effectuer des opérations arithmétiques courantes, telles que l'addition, la soustraction, la multiplication, etc. Voici une liste complète des opérateurs arithmétiques de PHP :

Opérateur	La description	Exemple	Résultat
+	Une addition	$x + $y	Somme de $x et $y
-	Soustraction	$x - $y	Différence de $x et $y.
*	Multiplication	$x * $y	Produit de $x et $y.
/	Division	$x / $y	Quotient de $x et $y
%	Module	$x % $y	Reste de $x divisé par $y

L'exemple dans le slide suivant vous montrera ces opérateurs arithmétiques en action :

```php
<?php
$x = 10;
$y = 4;

echo($x + $y);
echo "<br>";  //affiche 14

echo($x - $y);
echo "<br>";    //affiche 6

echo($x * $y);
echo "<br>";    //affiche 40

echo($x / $y);
echo "<br>";    //affiche 2.5

echo($x % $y); //affiche 2
?>
```

Opérateurs d'affectation PHP

Les opérateurs d'affectation sont utilisés pour affecter des valeurs aux variables.

Opérateur	La description	Exemple	Est le même que
=	Attribuer	$x = $y	$x = $y
+=	Ajouter et attribuer	$x += $y	$x = $x + $y
-=	Soustraire et assigner	$x -= $y	$x = $x - $y
*=	Multiplier et attribuer	$x *= $y	$x = $x * $y
/=	Diviser et attribuer un quotient	$x /= $y	$x = $x / $y
%=	Diviser et attribuer le module	$x %= $y	$x = $x % $y

L'exemple dans le slide suivant vous montrera ces opérateurs d'affectation en action :

```php
<?php
$x = 10;
echo $x;
echo "<br>";

$x = 50;
$x /= 10;
echo $x;
echo "<br>";

$x = 100;
$x %= 15;
echo $x;
?>
```

Opérateurs de comparaison PHP

Les opérateurs de comparaison permettent de comparer deux valeurs de manière booléenne.

Opérateur	Nom	Exemple	Résultat
==	Égal	$x == $y	Vrai si $x est égal à $y
===	Identique	$x === $y	Vrai si $x est égal à $y et qu'ils sont du même type
!=	Inégal	$x != $y	Vrai si $x n'est pas égal à $y
<>	Inégal	$x <> $y	Vrai si $x n'est pas égal à $y
!==	Pas identique	$x !== $y	Vrai si $x n'est pas égal à $y, ou s'ils ne sont pas du même type
<	Moins que	$x < $y	Vrai si $x est inférieur à $y
>	Plus grand que	$x > $y	Vrai si $x est supérieur à $y
>=	Plus grand ou égal à	$x >= $y	Vrai si $x est supérieur ou égal à $y
<=	Inférieur ou égal à	$x <= $y	Vrai si $x est inférieur ou égal à $y

```php
<?php
$x = 25;
$y = 35;
$z = "25";

var_dump($x == $z);
echo "<br>";

var_dump($x === $z);
echo "<br>";

var_dump($x > $y);
echo "<br>";

var_dump($x <= $y);
echo "<br>";

var_dump($x >= $y);
?>
```

Opérateurs d'incrémentation et de décrémentation PHP

Les opérateurs d'incrémentation/décrémentation sont utilisés pour incrémenter/décrémenter la valeur d'une variable.

Opérateur	Nom	Effet
++$x	Pré-incrémentation	Incrémente $x de un, puis renvoie $x
$x++	Post-incrémentation	Renvoie $x, puis incrémente $x de un
--$x	Pré-décrémentation	Décrémente $x de un, puis retourne $x
$x--	Post-décrémentation	Renvoie $x, puis décrémente $x de un

L'exemple dans le slide suivant vous montrera ces opérateurs d'incrémentation et de décrémentation en action :

```php
<?php
$x = 10;
echo ++$x;
echo "<br>";
echo $x;
echo "<hr>";

$x = 10;
echo $x--;
echo "<br>";
echo $x;
?>
```

Opérateurs logiques PHP

Les opérateurs logiques sont généralement utilisés pour combiner des instructions conditionnelles.

Opérateur	Nom	Exemple	Résultat
and	Et	$x and $y	Vrai si $x et $y sont vrais
or	Ou	$x or $y	Vrai si $x ou $y est vrai
xor	Xou	$x xor $y	Vrai si $x ou $y est vrai, mais pas les deux
&&	Et	$x && $y	Vrai si $x et $y sont vrais
||	Ou	$x || $y	Vrai si $x ou $y est vrai
!	Pas	!$x	Vrai si $x n'est pas vrai

L'exemple suivant vous montrera ces opérateurs logiques en action :

```php
<?php
$year = 2022;
// Les années bissextiles sont divisibles par 400 ou par 4, mais pas par 100.
if(($year % 400 == 0) || (($year % 100 != 0) && ($year % 4 == 0))){
    echo "$year est une année bissextile.";
} else{
    echo "$year n'est pas une année bissextile.";
}
?>
```

Opérateurs de chaîne PHP

Il existe deux opérateurs spécialement conçus pour les chaînes .

Opérateur	La description	Exemple	Résultat
.	Enchaînement	$str1 . $str2	Concaténation de $str1 et $str2
.=	Affectation de concaténation	$str1 .= $str2	Ajoute le $str2 au $str1

L'exemple suivant vous montrera ces opérateurs de chaîne en action :

```php
<?php
$x = "Hello";
$y = " World!";
echo $x . $y; // Outputs: Hello World!
$x .= $y;
echo $x; // Outputs: Hello World!
?>
```

Trouver la valeur absolue d'un nombre

La valeur absolue d'un entier ou d'un flottant peut être trouvée avec la fonction abs(), comme illustré dans l'exemple suivant :

```php
<?php
echo abs(5) . "<br>";    // 0utputs: 5 (integer)
echo abs(-5) . "<br>";   // 0utputs: 5 (integer)
echo abs(4.2) . "<br>";  // 0utputs: 4.2 (double/float)
echo abs(-4.2) . "<br>"; // 0utputs: 4.2 (double/float)
?>
```

Comme vous pouvez le voir si le nombre donné est négatif, la valeur renvoyée est positive. Mais, si le nombre est positif, cette fonction renvoie simplement le nombre.

Arrondir une valeur fractionnaire vers le haut ou vers le bas

La fonction ceil() peut être utilisée pour arrondir une valeur fractionnaire à la valeur entière la plus élevée suivante, tandis que la fonction floor() peut être utilisée pour arrondir une valeur fractionnaire à la valeur entière la plus basse suivante, comme illustré dans l'exemple suivant :

```php
<?php
// Arrondir les fractions
echo ceil(4.2) . "<br>";    // 0utputs: 5
echo ceil(9.99) . "<br>";   // 0utputs: 10
echo ceil(-5.18) . "<br>";  // 0utputs: -5

// Arrondir les fractions à la baisse
echo floor(4.2) . "<br>";    // 0utputs: 4
echo floor(9.99) . "<br>";   // 0utputs: 9
echo floor(-5.18) . "<br>";  // 0utputs: -6
?>
```

Trouver la racine carrée d'un nombre

Vous pouvez utiliser la fonction sqrt() pour trouver la racine carrée d'un nombre positif. Cette fonction renvoie une valeur spéciale NAN pour les nombres négatifs. Voici un exemple :

```php
<?php
echo sqrt(9) . "<br>";   // 0utputs: 3
echo sqrt(25) . "<br>";  // 0utputs: 5
echo sqrt(10) . "<br>";  // 0utputs: 3.1622776601684
echo sqrt(-16) . "<br>"; // 0utputs: NAN
?>
```

Générer un nombre aléatoire

La fonction rand() peut être utilisée pour générer un nombre aléatoire. Vous pouvez éventuellement spécifier une plage en transmettant les arguments min, max, comme illustré dans l'exemple suivant :

```php
<?php
// Générer des nombres aléatoires
echo rand() . "<br>";
echo rand() . "<br>";

// Générer des nombres aléatoires entre 1 et 10 (inclus)
echo rand(1, 10) . "<br>";
echo rand(1, 10) . "<br>";
?>
```

Si la fonction rand() est appelée sans les arguments , facultatifs min, max elle renvoie un nombre pseudo-aléatoire entre 0 et getrandmax(). La fonction getrandmax() affiche la plus grande valeur aléatoire possible, qui n'est que de **2147483647** sur la plate-forme Windows. Ainsi, si vous avez besoin d'une plage supérieure à 32767, vous pouvez simplement spécifier les arguments min et max