codecademy

# Apprendre les variables PHP

## Analyser des variables dans des chaînes PHP

En PHP, les variables peuvent être analysées dans des chaînes spécifiées avec des guillemets doubles ( `"` ). Cela signifie que dans la chaîne, l'ordinateur remplacera une occurrence d'une variable par la valeur de cette variable.
Lorsque des caractères d'identification valides supplémentaires (c'est-à-dire des caractères pouvant être inclus dans un nom de variable) sont destinés à apparaître à côté de la valeur de la variable, le nom de la variable peut être entouré d'accolades `{}` , évitant ainsi toute confusion quant au nom de la variable.

```
$my_var = "cat";

echo "There is one $my_var on the mat";

/* If there were three cats, then you can
type: */
echo "There are three {$my_var}s on the
mat ";
/* The curly braces help to avoid
confusion between the variable name and
the letter s, so PHP does not consider
the variable name as my_vars */
```

## Réaffectation des variables PHP

En PHP, les variables sont affectées de valeurs avec l'opérateur d' *affectation* `=` ( ). La même variable peut ensuite être réaffectée à une nouvelle valeur en utilisant le même opérateur.
Ce processus est connu sous le nom de *réaffectation* .

```
$var1 = "Bob";
echo $var1;
// var1 holds the value "Bob"

$var1 = "John";
echo $var1;
// var1 now holds the value "John"
```

## Concaténer des chaînes en PHP

En PHP, si vous souhaitez joindre deux chaînes ensemble, vous devez utiliser l' `.` opérateur.
Ce processus est appelé *concaténation* . Placez l' `.` opérateur entre les deux chaînes afin de les joindre.
Notez que les chaînes sont jointes telles quelles, sans insérer de caractère d'espacement. Donc, si vous avez besoin de mettre des espaces, vous devez incorporer l'espace manuellement dans la chaîne.

```
echo "Hello,"." welcome to Codecademy!";
// prints - Hello, welcome to Codecademy!
```

## Ajouter une chaîne en PHP

En PHP, il existe un raccourci pour ajouter une nouvelle chaîne à la fin d'une autre chaîne. Cela peut être facilement fait avec l' *opérateur d'affectation de concaténation de chaînes* ( `.=` ).
Cet opérateur ajoutera la valeur à sa droite à la valeur à sa gauche, puis réaffectera le résultat à la variable à sa gauche.

```
$str = "Hello, ";
$str .= "World!";

echo $str;
// Output: Hello, World!
```

## Chaînes PHP

En PHP, une *chaîne* est une séquence de caractères entourée de guillemets doubles. Il peut être aussi long que vous le souhaitez et contenir des lettres, des chiffres, des symboles et des espaces.

```
echo "Hello 123"; // prints Hello 123
```

## PHP copie des variables

En PHP, la valeur d'une variable peut être assignée à une autre variable.
Cela crée une copie de la valeur de cette variable et lui attribue le nouveau nom de variable.
Les modifications apportées à la variable d'origine n'affecteront pas la copie et les modifications apportées à la copie n'affecteront pas l'original. Ces variables sont des entités entièrement distinctes.

```
$original = "Ice T";
$copy = $original;
$orginal = "Iced Tea";
echo $copy; // "Ice T";
```

## Séquences d'échappement de chaîne PHP

En PHP, les caractères spéciaux doivent parfois être échappés afin de les inclure dans une chaîne. *Les séquences d'échappement* commencent par une barre oblique inverse ( `\` ).
Il existe une variété de séquences d'échappement qui peuvent être utilisées pour accomplir différentes tâches. Par exemple, pour inclure une nouvelle ligne dans une chaîne, la séquence `\n` peut être utilisée.
Pour inclure des guillemets doubles, la séquence `\"` peut être utilisée. De même, pour inclure des guillemets simples, la séquence `\'` peut être utilisée.

```
echo "Hello, World!\nThis is a String!";
/* prints-
Hello, World!
This is a String!
*/
```

## Ordre d'évaluation PHP pendant l'affectation

En PHP, lorsqu'une affectation a lieu, les opérations à droite de l'opérateur d'affectation ( `=` ) seront d'abord évaluées à une seule valeur. Le résultat de ces opérations sera alors affecté à la variable.

```
$var1 = 5 + 6 / 2;
/* Here, the operations to the right of
the assignment operator will be carried
out first. So first 6 will be divided by
2 (6 / 2 = 3). Then 3 will be added to
5 (5 + 3 = 8). Finally, the value 8 will
be assigned to $var1. */

echo $var1;
// Output: 8
```

## Variables PHP

En PHP, les variables sont affectées de valeurs avec l'opérateur d'affectation ( `=` ).
Les noms de variables peuvent contenir des chiffres, des lettres et des traits de soulignement ( `_` ). Un *sigil* ( `$` ) doit toujours précéder un nom de variable. Ils ne peuvent pas commencer par un nombre et ils ne peuvent pas avoir d'espaces ou de caractères spéciaux.
La convention en PHP est d'utiliser la casse du *serpent* pour nommer les variables ; cela signifie que les mots en minuscules sont délimités par un caractère de soulignement ( `_` ). Les noms de variables sont sensibles à la casse.

```
$my_variable = "Hello";

$another_cool_variable = 25;
```

## Opérateur d'affectation de référence PHP

En PHP, l' *opérateur d'affectation de référence* ( `=&` ) est utilisé pour créer une nouvelle variable en tant qu'alias vers un emplacement existant en mémoire.
En d'autres termes, l'opérateur d'affectation de référence ( `=&` ) crée deux noms de variables qui pointent vers la même valeur. Ainsi, les modifications apportées à une variable affecteront l'autre, sans avoir à copier les données existantes.

```
$var1 = 5;
$var2 =& $var1;

$var1 = 6;
echo $var2;
// Output: 6
```

## Valeurs entières en PHP

PHP prend en charge les valeurs *entières* pour les nombres.
Les nombres entiers sont l'ensemble de tous les nombres entiers, leurs contreparties négatives et zéro. Autrement dit, un *entier* est un nombre de l'ensemble $\mathbb{Z}$ = {..., -2, -1, 0, 1, 2, ...}.

## Opérateur d'exponentiation en PHP

PHP prend en charge un opérateur arithmétique pour *l'exponentiation* ( `**` ).
Cet opérateur donne le résultat de l'élévation de la valeur de gauche à la puissance de la valeur de droite.

```
$n = 4;
echo 2 ** 4;
// Output: 16
```

## Opérateurs arithmétiques en PHP

PHP prend en charge *les opérateurs arithmétiques* pour l'addition ( `+` ), la soustraction ( `-` ), la multiplication ( `*` ) et la division ( `/` ).
Les opérateurs PHP renverront des *entiers* chaque fois que le résultat d'une opération est évalué à un nombre entier. Si le résultat donne une fraction ou un nombre décimal, il renverra un *nombre à virgule flottante* .

```
$a = 6 / 3;
// The variable $a will hold an integer value, since the operation evaluates to a whole number.

$b = 7 / 3;
// The variable $b will hold a floating point value, since the operation evaluates to a decimal number.
```

## L'opérateur modulo

PHP prend en charge un *opérateur modulo* ( `%` ). L' *opérateur modulo* renvoie le reste de l'opérande gauche divisé par l'opérande droit. Les opérandes d'une opération modulo sont convertis en nombres entiers avant d'effectuer l'opération. L'opération renvoie un entier avec le même signe que le dividende.

```
$a = 9 % 2.3;
//2.3 is converted to the integer, 2. The remainder of 9 % 2 is 1. So the variable $a will hold the integer value 1.

$b = -19 % 4;
//The remainder of this operation is -3. So the variable $b will hold the integer value -3.

$c = 20 % 2;
//The remainder of this operation is 0. So the variable $c will hold the integer value 0.
```

## Nombres à virgule flottante en PHP

PHP prend en charge les *nombres à virgule flottante* (décimaux). Ils peuvent être utilisés pour représenter des quantités fractionnaires ainsi que des mesures précises. Quelques exemples de nombres à virgule flottante sont `1.5` , `4.231` , `2.0` , etc.