



6) Les Opérateurs logique et de comparaison

Maintenant nous allons parler des opérateurs logiques et de comparaison. On peut bien entendu avoir des opérations un peu plus complexes. Voici des tableaux récapitulatifs des opérateurs (sources : <http://www.codingame.com/>).

Le tableau ci-dessous reprend la liste des opérateurs logiques par ordre de priorité.

EXEMPLE	NOM	RÉSULTAT
! \$a	Not (Non)	TRUE si \$a n'est pas TRUE.
\$a && \$b	And (Et)	TRUE si \$a ET \$b sont TRUE.
\$a \$b	Or (Ou)	TRUE si \$a OU \$b est TRUE.
\$a and \$b	And (Et)	TRUE si \$a ET \$b valent TRUE.
\$a xor \$b	XOR	TRUE si \$a OU \$b est TRUE, mais pas les deux en même temps.
\$a or \$b	Or (Ou)	TRUE si \$a OU \$b valent TRUE.

Les opérateurs de comparaison, comme leur nom l'indique, vous permettent de comparer deux valeurs.

EXEMPLE	NOM	RÉSULTAT
\$a == \$b	Egal	TRUE si \$a est égal à \$b après le transtypage.
\$a === \$b	Identique	TRUE si \$a est égal à \$b et qu'ils sont de même type.
\$a != \$b	Différent	TRUE si \$a est différent de \$b après le transtypage.
\$a <> \$b	Différent	TRUE si \$a est différent de \$b après le transtypage.
\$a !== \$b	Différent	TRUE si \$a est différent de \$b ou bien s'ils ne sont pas du même type.
\$a < \$b	Plus petit	TRUE si \$a est strictement plus petit que \$b.
\$a > \$b	Plus grand	TRUE si \$a est strictement plus grand que \$b.
\$a <= \$b	Inférieur ou égal	TRUE si \$a est plus petit ou égal à \$b.
\$a >= \$b	Supérieur ou égal	TRUE si \$a est plus grand ou égal à \$b.

Vous remarquerez que la plupart des opérateurs vous les connaissez déjà grâce au cours d'algo, javascript ou MySQL que vous avez eu/ou que vous aurez.

Faisons un exemple de comparaison plus complexe. On va demander à l'utilisateur quelle heure il est et on va essayer de voir est ce que le magasin est ouvert sachant que le magasin ouvre **entre 9h et 13h et ensuite entre 14h et 18h**. Ce sont leurs horaires d'ouverture.

Créons un nouveau fichier qu'on appellera « **opérateurs.php** ». Vous aurez remarqué qu'à chaque nouveau chapitre, on créera un fichier en son nom, comme ça pour réviser tel sujet, vous ouvrirez les fichiers correspondants.

Dans ce fichier opérateurs on mettra ceci :

```
<?php
    $heure = (int)readline('Entrez une heure : ');
    if(($heure > 8 AND $heure <= 12) || ($heure >= 14 && $heure < 18)){
        echo "Le magasin sera ouvert à $heure heures";
    }else{
        echo "Le magasin sera fermé à " . $heure . " heures";
    }
?>
```

Testez, vous verrez qu'il affichera le message qui convient en fonction de l'heure vous indiquerez.

```
PS C:\wamp64\www\coursphp> php opérateurs.php
Entrez une heure : 9
Le magasin sera ouvert à 9 heures
PS C:\wamp64\www\coursphp> php opérateurs.php
Entrez une heure : 8
Le magasin sera fermé à 8 heures
PS C:\wamp64\www\coursphp>
```

Vous remarquerez que dans le code j'ai fait exprès d'écrire différemment le et, une fois en lettre **AND** et une fois avec le signe **&&**. Pareil j'ai aussi mis les deux manières d'affichage avec la concaténation et sans. Pour le ou j'ai utilisé les doubles pipes **||** mais j'aurai bien entendu pu mettre le **OR**. A vous de choisir mais bien sur que dans votre code vous utiliserez toujours la même manière que vous aurez choisi pour rester cohérent.

Pour mieux comprendre comment ça fonctionne, il faut savoir que chaque partie de la condition donne un booléen qui est soit vrai soit faux en fonction de la condition.

Petit rappel :

VRAI && VRAI = VRAI

VRAI && FAUX = FAUX

FAUX && FAUX = FAUX

VRAI || VRAI = VRAI

VRAI || FAUX = VRAI

FAUX || FAUX = FAUX

(Exo)

- 1) Si vous avez bien compris tous les opérateurs je veux que vous repreniez l'exemple sur les heures d'ouvertures du magasin. Je veux que vous inversiez l'affichage. Que doit-on mettre dans le IF pour que nos horaires ne changent pas. (Il y a une astuce simple)

```
if(){  
    echo "Le magasin sera fermé à " . $heure ." heures";  
}else{  
    echo "Le magasin sera ouvert à $heure heures";  
}
```

- 2) On va simuler une entrée à la discothèque.

Créez une variable users1 qui recevra un tableau associatif avec les clés :

firstname Will
lastname Smith
sexe M
age 53
vaccinated true

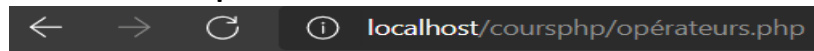
Créez une variable users2 qui recevra un tableau associatif avec les clés :

firstname Lara
lastname Croft
sexe F
age 17
vaccinated false

Créez une variable users3 qui recevra un tableau associatif avec les clés :

firstname Marion
lastname Cotillard
sexe F
age 46
vaccinated true

Créez une condition qui vérifie si vous êtes une personne majeure et vacciné. Si c'est le cas ça affichera « Bienvenue dans le club firstname lastname ! ». Ensuite en fonction que c'est une femme ou un homme. Il **affichera à la ligne** : « Prenez un bracelet bleu » pour les hommes et « Prenez un bracelet rose » pour les femmes.



localhost/coursphp/opérateurs.php

Bienvenue dans le club Will Smith!
Prenez un bracelet bleu



localhost/coursphp/opérateurs.php

Bienvenue dans le club Marion Cotillard!
Prenez un bracelet rose

7) Les Boucles

Nous allons parler des boucles en PHP. Vous les avez je pense déjà tous vu dans d'autres langages mais je vais y revenir dessus. En PHP on va voir trois types de boucle :

- Tant que (while)
- Pour(for)
- Pour chaque (foreach)

La boucle WHILE

Syntaxe :

- while(condition(s)){
 faire instruction(s) ;
}

C'est donc le **tant que**. Il fera **une** ou **plusieurs instructions tant que** la **condition** est **valide**. Créons un nouveau fichier PHP que nous appellerons **boucles.php**, dans ce fichier mettez ce code :

```
<?php
$nombre=(int)readline("Entrez un nombre (entrez 0 pour arrêter) : ");
while($nombre != 0){
    echo "Voici votre nombre : ". $nombre . "\n";
    $nombre=(int)readline("Entrez un nombre (entrez 0 pour arrêter) : ");
}
?>
```

Exécutez le code via le terminal. Vous verrez que vous pourrez mettre des nombres à l'infini tant que ce n'est pas zéro que vous mettez.

Un autre exemple :

```
$nombre=(int)readline("Entrez un nombre entre 0 et 10 pour gagner un lot : ");
$numéroGagnant = rand(0,10);
while($nombre != $numéroGagnant){
    echo "Mauvais numéro, vous n'avez pas gagné !". "\n";
    $nombre=(int)readline("Retentez votre chance, entrez un nombre à nouveau : ");
}
echo "Bravo !!! \nVous avez enfin trouvé le numéro gagnant ! \nC'était le numéro $numéroGagnant";
```

Exécutez ce code via le terminal. Ce code va vérifier si on tombe sur un numéro au hasard entre 0 et 10 compris. Quand on tombe dessus on a gagné. Utilisation de fonction **rand(min,max)** qui va retourner un entier.

Vous voyez c'est simple d'utilisation. Il faut faire attention à ce que dans les instructions, il y a une partie de la condition qui puisse changer. Parce que si on ne met pas de changement, ça sera une boucle infinie. Pour **quitter** une **boucle infinie** vous pouvez faire au clavier **CRTL+C**.

(Exo)

- 1) Créez-moi une application qui demande d'écrire un mot au clavier, tant qu'on n'écrit pas « stop », il continuera de demander d'écrire une chaîne de caractère.

```
PS C:\wamp64\www\coursphp> php boucles.php
Entrez un mot : julien
Entrez un mot : biensur
Entrez un mot : oklm
Entrez un mot : stop
Vous avez quitté le programme !!!
PS C:\wamp64\www\coursphp>
```

La boucle FOR

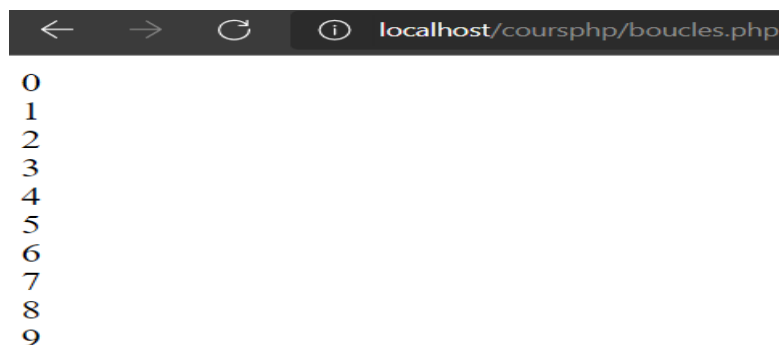
Syntaxe :

```
- for(variableInitial ; condition ; evolutionVariableInitial){
    instruction(s)
}
```

La boucle « for » sert principalement à parcourir. Prenons un exemple si j'ai envie d'afficher les 10 premiers chiffres. Je peux faire :

```
for($i=0; $i < 10;$i++){
    echo $i . "<br>";
}
```

Ce qui donne sur le navigateur :



```
<  >  ↻  ⓘ localhost/coursphp/boucles.php
0
1
2
3
4
5
6
7
8
9
```

Il va tout d'abord initialiser la valeur de \$i à zéro, si on voulait commencer à 1 on aurait mis cette valeur à un. Ensuite je mets une condition qui va permettre de sortir de cette boucle : tant que c'est plus petit que 10, écrit moi la valeur de \$i. Ensuite une fois qu'il a écrit la valeur, il va incrémenter le \$i de 1 (\$i++ ; c'est l'équivalent de \$i = \$i+1 ;).

On peut bien entendu incrémenter comme on veut et aussi décrémenter comme on le souhaite. A savoir qu'écrire $\$i = \$i + 2$; c'est équivalent de $\$i += 2$;

On peut aussi grâce à la boucle FOR, afficher le contenu d'un tableau. Reprenons notre tableau de notes :

```
$notes = [18,13,5,9,10];
```

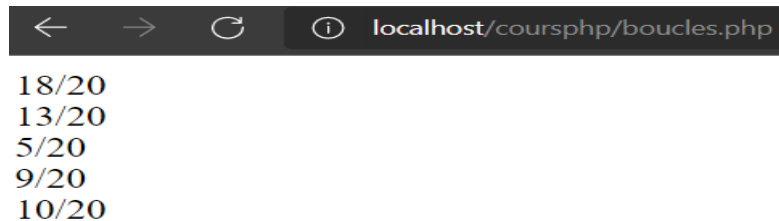
Pour afficher toutes les notes il ne faudra plus faire « echo \$notes[0] \$notes[1] \$notes[2] \$notes[3] \$notes[4] ; » mais plutôt utiliser une boucle qui nous le fera automatiquement.

Ici dans notre exemple de 5 notes ça va encore de les afficher un par un, mais si on avait 5000 notes ça aurait été plus compliqué.

On va utiliser le for :

```
for ($i=0; $i < sizeof($notes); $i++) {  
    echo $notes[$i]. "/20 <br>";  
}
```

Ce qui donnera dans vos navigateurs :

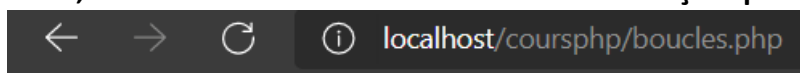


```
18/20  
13/20  
5/20  
9/20  
10/20
```

Il va répéter le nombre de fois qu'il y a d'élément dans le tableau. Ici pour savoir le nombre d'élément que contient un tableau on appelle la fonction PHP **sizeof(nomTableau)**, dans ce cas il va retourner 5. Je vais vous dessiner ce qu'il se passe étape par étape.

(Exos)

- 1) Reprenez le tableau des jours de la semaine et affichez-moi les jours de la semaine.
- 2) Créez un tableau « mois » qui représentera les mois de l'année : Janvier, février, Mars,... et affichez moi tous les 2 mois en commençant par février :

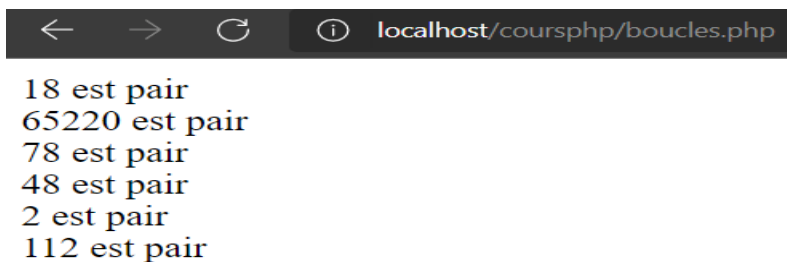


```
Février Avril Juin Aout Octobre Décembre
```

- 3) Prenez ce tableau ci-dessous (copier-coller le)

```
$nombre = [11,18,99,17,65220,6485,78,97,48,2,112];
```

Affichez-moi tous les nombres pairs de ce tableau de cette manière :



```
18 est pair  
65220 est pair  
78 est pair  
48 est pair  
2 est pair  
112 est pair
```

La boucle FOREACH

Syntaxe :

```
- foreach(listeOuTable as nomVariable){  
    instruction(s);  
}
```

Prenons un exemple concret, reprenons notre fameux tableau de note. Le foreach va donc prendre ce tableau \$notes et le mettre dans une variable \$note qu'on affichera pour chaque élément. Pour chaque élément du tableau \$notes, affiche-les-moi dans la variable \$note.

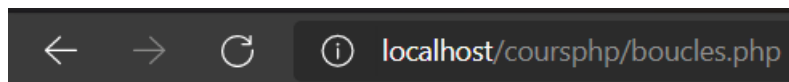
```
$notes = [18,13,5,9,10];  
foreach ($notes as $note) {  
    echo $note . " ";  
}
```

Vous n'aurez donc plus besoin de déclarer une variable d'initialisation qui sert d'indice (\$i), ni même de mettre une condition pour savoir le nombre d'élément ni le besoin d'incrémenter cette variable. Il fera tout ça tout seul.

On peut aussi utiliser cette boucle pour récupérer la clé d'un tableau indexé. Reprenez le tableau associatif qu'on avait fait en **enlevant les notes**.

```
$users = [  
    'firstname' => 'Lara',  
    'lastname' => 'Croft',  
    'sexe' => 'F',  
    'dateOfBirth' => "23/10/1995",  
    'city' => 'London'  
];  
foreach ($users as $key => $element) {  
    echo $key . ' : ' . $element . '<br>';  
}
```

Ce qui affichera :



```
firstname : Lara  
lastname : Croft  
sexe : F  
dateOfBirth : 23/10/1995  
city : London
```

Vous comprenez que ça peut nous faciliter la vie pour explorer des données complexe.

Pour les plus costaud, je vais cette fois ci reprendre le tableau d'origine donc avec le tableau de notes dedans.

```
$users = [
    'firstname' => 'Lara',
    'lastname' => 'Croft',
    'gender' => 'F',
    'dateOfBirth' => "23/10/1995",
    'notes' => [18,13,5,10,9],
    'city'=> 'London'
];
foreach ($users as $key => $element) {
    if (is_array($element)){
        echo $key.' : [';
        foreach($element as $note){
            echo $note. "/20 ";
        }
        echo "]<br>";
    }else{
        echo $key . ' : ' . $element . '<br>';
    }
}
```

Ce qui affichera :

```
< > ↻ ⓘ localhost/coursphp/boucles.php

firstname : Lara
lastname : Croft
sexe : F
dateOfBirth : 23/10/1995
notes : [18/20 13/20 5/20 10/20 9/20 ]
city : London
```

Ici on doit faire une boucle dans une boucle parce qu'on doit aussi rentrer dans le tableau de notes.

(Exos)

- 1) Reprenez votre tableau des mois et essayez de me les afficher en les séparant par des tirets. En utilisant le foreach bien entendu.

```
< > ↻ ⓘ localhost/coursphp/boucles.php

Janvier - Février - Mars - Avril - Mai - Juin - Juillet - Aout - Septembre - Octobre - Novembre - Décembre
```

- 2) Reprenez le tableau de notes et pour chaque note je veux qu'on affiche un message en fonction de la note obtenu. Attention 3 messages différents.

```
< > ↻ ⓘ localhost/coursphp/boucles.php

Tu as réussi avec la note de 18/20
Tu as réussi avec la note de 13/20
Tu as raté avec la note de 5/20
Tu as raté avec la note de 9/20
Tu as eu tout pile la moitié 10/20
```


(Exo **DIFFICILE** sur le chapitre des boucles pour ceux qui sont en TGV)

- 1) Je vais vous demander de créer une application PHP qui reçoit au clavier des mots et place ces mots dans un tableau. Pour arrêter l'ajout des mots, il suffira de taper au clavier « stop ». Je veux ensuite que vous m'affichiez ces mots qui sont dans ce tableau séparé par des virgules. (Astuce pour déclarer un tableau vide : `$tab = [] ;`)
(astuce pour ajouter une variable à un tableau : `$tab[] = variable ;`)

```
PS C:\wamp64\www\coursphp> php boucles.php
Entrez un nouveau mot ou taper "stop" pour arreter : bonjour
Entrez un nouveau mot ou taper "stop" pour arreter : bonsoir
Entrez un nouveau mot ou taper "stop" pour arreter : salut
Entrez un nouveau mot ou taper "stop" pour arreter : salutations
Entrez un nouveau mot ou taper "stop" pour arreter : hello
Entrez un nouveau mot ou taper "stop" pour arreter : bonne nuit
Entrez un nouveau mot ou taper "stop" pour arreter : stop
bonjour, bonsoir, salut, salutations, hello, bonne nuit,
```

- 2) Reprenez ce tableau qu'on a déjà vu dans le chapitre sur les tableaux.

```
$classe = [
    [
        'firstname' => "Julien",
        'lastname' => "Dunia",
        'notes' => [8, 15, 12]
    ],
    [
        'firstname' => "Hakima",
        'lastname' => "Darmouch",
        'notes' => [18, 5, 10]
    ],
    [
        'firstname' => "Christian",
        'lastname' => "Bale",
        'notes' => [7, 19, 5]
    ]
];
```

Je veux que vous m'affichiez les éléments de cette manière en utilisant tout ce que je vous ai montré sur le chapitre des boucles (astuce pour la moyenne je veux aussi une boucle) :

```
< > ↺ ⓘ localhost/coursphp/boucles.php
```

```
firstname : Julien
lastname : Dunia
notes : Moyenne 11.6666666666667
```

```
firstname : Hakima
lastname : Darmouch
notes : Moyenne 11
```

```
firstname : Christian
lastname : Bale
notes : Moyenne 10.3333333333333
```