



## 4) Les Tableaux

Dans ce chapitre nous allons parler des tableaux. Les tableaux peuvent être utiles par exemple pour sauvegarder des notes ou pour classer tout autres choses. Les tableaux sont souvent associés aux boucles pour justement le remplir ou le traiter. Ils permettent de sauvegarder une liste de valeur. Cette liste peut être indexé ou non, on verra la différence.

Pour créer un tableau c'est simple. **Créez** un fichier **tableau.php** où vous introduirez ce bout de code.

```
<?php
    $notes = [18,13,5,9,10];
?>
```

On peut indiquer plusieurs valeurs, chacune d'elles devra être séparé par des virgules. On peut y mettre n'importe quel type que l'on a vu précédemment. Vous pouvez même avoir des tableaux dans des tableaux. **ATTENTION** les indices d'un tableau en PHP **commencent à ZERO**.

Ici nous avons donc 5 valeurs dans notre tableau. Si je veux afficher la 3<sup>ème</sup> note de ce tableau, je dois faire :

```
<?php
    $notes = [18,13,5,9,10];
    echo $notes[2];
?>
```

Vous pouvez vérifier le résultat sur votre page web ou via le terminal.

Il y a aussi l'ancienne méthode pour créer des tableaux avec le mot clé **array**.

```
<?php
// $notes = [18,13,5,9,10];
$notes = array(18,13,5,9,10);
echo $notes[2];
?>
```

Vous pouvez tester, vous verrez que le résultat sera le même. Cette ancienne méthode est encore valable aujourd'hui mais est de moins en moins utilisée.

**(Exos) Mettez en commentaire votre echo pour passer aux exercices suivants.**

- 1) Prenez le tableau « \$notes » de mon exemple. Vous m'afficherez la moyenne dans une variable et la somme de toutes les notes dans une autre. Le message donnera :  
C'est la moyenne : moyenne  
C'est la somme : somme
- 2) Créez-moi un tableau « animal » qui contient « chien, chat, éléphant et tortue ». Affichez moi le 2<sup>ème</sup> et 4<sup>ème</sup> élément de ce tableau.
- 3) Créez-moi un tableau des jours de la semaine. Affichez-moi ensuite le jour qu'on est aujourd'hui via ce tableau. Ainsi qu'avant-hier et après-demain.

Comme je l'ai dit plus haut, on peut aussi mettre un tableau dans un tableau. On appelle ce procédé un tableau à deux dimensions.

```
$tab = ["Jack", "Bauer", [18,13,5,9,10]];
```

Ici on met dans la variable « \$tab » un tableau contenant un tableau. Il représente les notes de Jack Bauer.

Si on veut par exemple récupérer sa 4<sup>ème</sup> note il faudra procéder de la manière suivante :

```
$tab = ["Jack", "Bauer", [18,13,5,9,10]];
echo $tab[2][3];
```

Ici il va prendre dans un premier temps le 3<sup>ème</sup> élément de son premier tableau, ensuite vu que ce troisième élément est un tableau lui-même, on va sélectionner le 4<sup>ème</sup> élément de ce tableau.

**(Exo)**

- 1) Créez moi tableau à deux dimensions ayant comme nom « tab2dim ». Donnez-lui comme valeur « James, Bond, M, 7/7/2007, [Lundi, Mardi, Mercredi, Jeudi, Vendredi, Samedi, Dimanche], London ». Affichez-moi ça. Attention jour de hier.

```
← → ↻ ⓘ localhost/coursphp/tableau.php
Nom : Bond
Prenom : James
Sexe : M
Date de naissance : 7/7/2007
Ville : London
Jour de hier : Lundi
```

## Les tableaux indexé/associatif

Nous avons jusqu'ici des tableaux indexés par défaut (commençant à 0). En termes de lisibilité, pour une autre personne, certaines choses ne sont pas claires. Maintenant nous allons voir les tableaux associatifs. On a la possibilité en PHP de donner des clés particulières à nos valeurs. Donc plutôt que d'utiliser un index qui est automatiquement défini 0, 1, 2, 3 etc. Vous pouvez utiliser des clés particulières, dans ce cas la notation sera un peu différente.

```
$users = [  
    'firstname' => 'Lara',  
    'lastname' => 'Croft',  
    'gender' => 'F',  
    'dateOfBirth' => "23/10/1995",  
    'notes' => [18,13,5,10,9],  
    'city' => 'London'  
];
```

On aura donc une clé firstname, une clé lastname, une clé notes etc.

Vous voyez qu'en terme de lisibilité, on y voit plus clair. Parce que pour afficher quelque chose du tableau on donnera le nom du champ qu'on veut afficher.

```
echo $users['city'];
```

Ici on voit bien qu'il affichera la ville. Et une personne externe pourrait le comprendre aussi. C'est plus lisible que de voir \$users[5].

Ici pour les notes c'est pareil, on saura qu'on cherche une note.

```
echo $users['notes'][1];
```

Il ira prendre la deuxième note du tableau de note.

C'est donc un tableau qui peut mieux gérer des données plus complexes.

On peut aussi changer les valeurs du tableau, il suffit de sélectionner le champ et de lui assigner une valeur.

```
$users["firstname"]="Edward";  
echo $users['firstname'];
```

On peut aussi ajouter une valeur, par exemple si je veux ajouter une 6<sup>ème</sup> note on peut faire :

```
$users['notes'][5]=20;
```

(Exos) Oubliez pas de mettre en commentaire ce que vous avez fait avant (// ou /\*..\*/)

- 1) Essayez de modifier le nom de famille en « Elric » et la 5<sup>ème</sup> note en lui donnant 17. Affichez-moi le nom, prénom et la 5<sup>ème</sup> note, en sautant à chaque fois à la ligne.
- 2) Créez un deuxième utilisateur avec un tableau indexé. Vous rajouterez un champ « joursDeSemaine » avec comme données un tableau des jours de la semaine. Affichez le nom, le prénom, la note moyenne ainsi que le jour de demain.
- 3) Ajoutez un huitième jour que vous appellerez « néant ».

Vous pouvez tester d'afficher le tableau. Avec le echo il risque de vous faire une erreur et de vous montrer juste « array », pour dire que c'est un tableau.

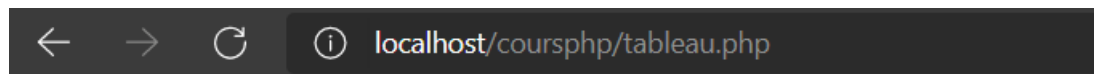
Il faudra alors utiliser la fonction print\_r() :

```
print_r($users['notes']);
```

Ce qui donnera le résultat ci-dessus :

```
PS C:\wamp64\www\coursphp> php tableau.php
Array
(
    [0] => 18
    [1] => 13
    [2] => 5
    [3] => 10
    [4] => 9
    [5] => 20
)
```

Ou via le navigateur :



Array ( [0] => 18 [1] => 13 [2] => 5 [3] => 10 [4] => 9 [5] => 20 )

On peut créer plusieurs utilisateurs en même temps, en créant un super tableau de tableau.

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

Ici si je veux afficher la 3<sup>ème</sup> note du deuxième utilisateur je dois taper :

```
echo $classeUser[1]['notes'][2];
```

(Exos) Oubliez pas de mettre en commentaire ce que vous avez fait avant (// ou /\*..\*/)

- 1) Essayez de m'afficher la 1<sup>ère</sup> note du 1<sup>er</sup> utilisateurs.
- 2) Essayez de m'afficher la 2<sup>ème</sup> note du troisième utilisateurs.
- 3) Essayez de m'afficher la moyenne du 3<sup>ème</sup> utilisateurs.

## 5) Les conditions

Jusqu'à maintenant on a créé des variables, y mettre des valeurs plus ou moins complexe à l'intérieur etc. Maintenant ce qu'on a envie de faire, c'est de mettre un peu de logique.

Les conditions vous nous permettent de vérifier si une valeur remplit tel ou tel condition etc.

**Créons** un nouveau fichier qu'on va appeler « **condition.php** ». Dedans on va créer une variable en lui donnant comme valeur 16.

### Le IF

Nous allons utiliser la condition **Si...Sinon « IF...ELSE »**

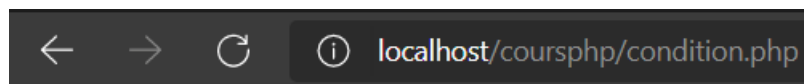
Voici sa syntaxe :

- Si (condition est vrai){
  - o Alors fais ceci
- }sinon{
  - o Fais cela
- }

Dans l'exemple je vais tester si ma note est plus grande que 10, si oui il indiquera comme message « Bien joué vous avez réussi » sinon il indiquera « C'est dommage, vous feriez mieux la prochaine fois ».

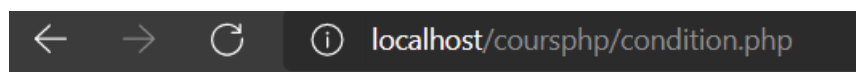
```
$note = 16;  
if ($note > 10){  
    echo "Bien joué vous avez réussi !";  
}else{  
    echo "C'est dommage, vous feriez mieux la prochaine fois";  
}
```

Ici on a donné 16 à notre note, du coup il retournera comme message



Bien joué vous avez réussi !

Vous aurez compris que si on change notre note en 8 par exemple cela donnera l'autre message.



C'est dommage, vous feriez mieux la prochaine fois

Voilà comment fonctionne le « IF ». Il y a aussi la possibilité d’imbriquer plusieurs IF l’un à la suite de l’autre.

```
if ($note >= 10) {  
    if ($note == 10){  
        echo "Vous avez tout juste la moyenne";  
    }else{  
        echo "Bien joué vous avez réussi !";  
    }  
}else{  
    echo "C'est dommage, vous feriez mieux la prochaine fois";  
}
```

Ici on veut donc faire en sorte que si la note est égale à 10, on lui envoie un message particulier. E

Il y a aussi la condition Si...sinon si...etc (IF...ELSE IF...ESLE). En gros on peut après un IF rajouter un autre IF.

```
if ($note > 10) {  
    echo "Bien joué vous avez réussi !";  
} else if($note == 0) {  
    echo "Tu es nul";  
}else{  
    echo "C'est dommage, vous feriez mieux la prochaine fois";  
}
```

Ici dans ce cas on veut rajouter que quand un utilisateur a une note de zéro, on ne lui affichera pas le même message que ceux qui ont raté.

C’est un peu pareil que de rajouter un IF dans un IF, le résultat est le même. A vous de choisir mais de manière générale les programmeurs en PHP utilisent le if elseif.

Vous pouvez dans PHP faire en sorte qu’un utilisateur introduit une valeur pour ensuite pouvoir la tester. C’est via la fonction « **readline()** ».

```
$note = readline('Entrez une note : ');
```

Voilà ce que ça peut donner

```
PROBLÈMES  SORTIE  TERMINAL  ...  
  
PS C:\wamp64\www\coursphp> php condition.php  
Entrez une note : 12  
Bien joué vous avez réussi !  
PS C:\wamp64\www\coursphp> |
```

**Attention**, à noter que ça ne marchera pas sur votre navigateur web. Il affichera juste « tu es nul » comme s'il avait reçu 0, vu qu'on n'a rien introduit.

**(Exos) Oubliez pas de mettre en commentaire ce que vous avez fait avant (// ou /\*..\*/)**

- 1) Créez une variable « âge » que vous donnerez comme valeur ce que vous voulez. Je veux qu'en fonction de l'âge que vous écrivez :  
si c'est plus de 18 ans, le message devra être « Vous êtes un adulte et vous avez « âge » ans. ». Si l'âge est de 18 ans vous envoyez comme message « Vous avez « âge » ans , bienvenu dans l'âge adulte » ». Si on a un âge en dessous de 18 ans vous enverrez comme message : « Vous avez « âge » ans et n'êtes pas encore un adulte » ».

- 2) Améliorez l'exercice 1, en faisant en sorte que l'utilisateur introduit un âge.

```
PS C:\wamp64\www\coursphp> php condition.php
Entrez votre âge : █
```

- 3) Créez une variable « genre » que vous initialisez soit en F, M ou X. Faites en sorte que quand c'est F on dit « vous êtes une femme », quand c'est M, on dit « vous êtes un homme » et quand c'est X « vous êtes un autre ».

- 4) Améliorez l'exercice 3, en faisant en sorte que l'utilisateur introduit le sexe. ET si on entre autres choses que ces 3 lettres il nous donne un message d'erreur.

```
PS C:\wamp64\www\coursphp> php condition.php
Veuillez introduire votre genre (M, F ou X) : █
```

**Différence entre == et ===.**

Pour utiliser une condition qui vérifie si une variable est égale à une valeur, jusqu'ici on a utilisé ==. Il faut savoir que le triple égale va vérifier le type ainsi que la valeur. Dans l'exemple avec les notes, j'ai mis cette fois ci le triple égale pour vérifier si la note qu'on introduit est égale à zéro.

```
$note = readline('Entrez votre note : ');
if ($note > 10) {
    echo "Bien joué vous avez réussi !";
} else if($note === 0) {
    echo "Tu es nul";
} else {
    echo "C'est dommage, vous feriez mieux la prochaine fois";
}
```

J'introduis 0

```
PS C:\wamp64\www\coursphp> php condition.php
Entrez votre note : 0
C'est dommage, vous feriez mieux la prochaine fois
PS C:\wamp64\www\coursphp> █
```

Il était censé me sortir le message « Tu es nul ». Il ne l'a pas fait parce que dans ce cas-ci, quand on fait un **readline()**, ce qu'on introduit au clavier est considéré comme chaîne de caractère. Du coup \$note va recevoir "0". Quand on le compare avec un ==, PHP va convertir

\$note en entier pour le comparer donc ça ira. Par contre ici si on utilise le ===, PHP va voir que l'origine du type \$note est une chaîne de caractère, c'est pour ça que ça ne passera pas. Pour pouvoir faire en sorte qu'il arrive à comparer avec ===, il faudra d'abord faire en sorte que \$note reçoive un entier.

Il suffira de caster readline() en entier, donc convertir directement ce qu'on écrit au clavier en entier avant qu'il arrive dans la variable \$note:

```
$note = (int)readline('Entrez votre note : ');
```

Et là vous pouvez tester ça marchera correctement. Il affichera « tu es nul » si on met 0.

## Le Switch

Le switch est une condition qui reçoit en paramètre une variable et prédéfinira en fonction de la valeur qu'il recevra, ce qu'il devra faire. C'est un peu comme le if...elseif. Prenons un exemple avec une action qu'on reçoit en paramètre et en fonction du chiffre choisi, il nous exécute ces actions :

```
$action = (int)readline("Entrez une action : ('1 : Attaquer, 2 : Defendre, 3 :  
Se soigner, 4 : Fuir, 5 : Ne rien faire) : ");  
if ($action === 1){  
    echo "Vous attaquez";  
}else if($action === 2){  
    echo "Vous défendez";  
}elseif($action === 3){  
    echo "Vous vous soignez";  
}elseif($action === 4){  
    echo "Vous fuyez";  
}elseif($action === 5){  
    echo "Vous ne faites rien";  
}else{  
    echo "Relancez le programme et Entrez une action 1,2,3,4 ou 5";  
}
```

Ici j'ai utilisé avec des si sinon si. Tout fonctionne parfaitement bien entendu.

```
PS C:\wamp64\www\coursphp> php condition.php  
Entrez une action : ('1 : Attaquer, 2 : Defendre, 3 : Se soigner, 4 : Fuir, 5 : Ne rien faire) : 1  
Vous attaquez  
PS C:\wamp64\www\coursphp> █
```

Le problème c'est qu'on se répète beaucoup, si j'avais 50 actions, ça serait vite lourd.

On va donc utiliser plutôt un switch pour tout ce qui est condition avec une liste prédéfinis.



La syntaxe c'est :

- switch(variable sur laquelle on va tester les valeurs) {
  - case (une valeur pour le premier cas) :
    - traitement à faire ;
    - break ;
  - case (deuxième valeur pour le second cas) :
    - traitement à faire ;
    - break ;
  - default :
    - traitement à faire si aucun cas ne correspond ;

Le break à chaque fois sert à sortir du switch s'il entre dans un cas.

Maintenant je vais vous montrer ce que donne le même programme avec action mais cette fois ci avec le switch :

```
switch($action){  
    case 1:  
        echo "Vous attaquez !";  
        break;  
    case 2:  
        echo "Vous défendez !";  
        break;  
    case 3:  
        echo "Vous vous soignez !";  
        break;  
    case 4:  
        echo "Vous fuyez !";  
        break;  
    case 5:  
        echo "Vous ne faites rien !";  
        break;  
    default:  
        echo "Relancez le programme et Entrez une action 1,2,3,4 ou 5";  
}
```

Ici on voit que c'est beaucoup plus lisible et compréhensible.

**(Exo) Oubliez pas de mettre en commentaire ce que vous avez fait avant (// ou /\*..\*/)**

- 1) Créez moi un programme qui m'affiche le jour de la semaine en fonction du numéro que vous introduisez (exemple si vous introduisez, 1 = Lundi, 2 = Mardi,... 7 = Dimanche)

```
PS C:\wamp64\www\coursphp> php condition.php  
Entrez le jour que nous sommes : 1  
Nous sommes Lundi !  
PS C:\wamp64\www\coursphp>
```