

Les tableaux en php

Table des matières

I. Contexte	3
II. Notions de base - syntaxe des différents types de tableaux	3
A. La syntaxe	3
B. Les tableaux indexés	4
C. Les tableaux associatifs	6
D. Exercice : Quiz	7
III. Manipuler les tableaux	9
A. Les manipulations de base	9
B. Les fonctions système des tableaux	11
C. Exercice : Quiz	13
IV. Les opérateurs de tableau	15
A. Introduction	15
B. Les opérateurs de comparaison	15
C. Opérateur arithmétique	16
D. Exercice : Quiz	16
V. Essentiel	17
VI. Auto-évaluation	18
A. Exercice	18
B. Test	18
Solutions des exercices	20

I. Contexte

Durée : 1 h

Environnement de travail : Visual Studio Code avec l'extension PHP IntelliSense

Pré-requis : aucun

Contexte

Le tableau (en anglais « *array* ») est un type de données structuré, qui permet de stocker plusieurs valeurs ensemble, regroupées sous un nom commun. Chaque valeur est associée à une clé qui permet d'y accéder. Cette clé peut-être implicite, il s'agit alors de l'index de position de la valeur dans la liste. La clé peut également être spécifiée, notamment quand il s'agit d'une chaîne de caractère. On parle alors de tableau associatif.

Ce type de tableau possède la même structure qu'un dictionnaire : chaque clé permet de retrouver facilement la valeur qui lui est associée. Un tableau peut lui-même comporter plusieurs tableaux. On parle alors de tableaux multidimensionnels. L'objectif de ce cours est d'apprendre à maîtriser la syntaxe de base des tableaux, à accéder aux données qu'ils contiennent, et comment les manipuler, notamment à l'aide des fonctions internes à PHP.

II. Notions de base - syntaxe des différents types de tableaux

A. La syntaxe

Il y a deux syntaxes différentes permettant d'initialiser, c'est-à-dire de créer, un tableau vide :

```
1 <?php
2 $firstArray = array();
3 $secondArray = [];
```

Un tableau contient des valeurs, auxquelles sont associées des clés. On parle de paire clé-valeur pour chaque élément d'un tableau. On peut initialiser un tableau qui contient déjà des paires clés-valeurs, en reprenant la même syntaxe que présentée plus haut. Un tableau peut être initialisé avec un nombre illimité de valeurs. Elles doivent être séparées par une virgule et présentées selon la syntaxe suivante : clé → valeur. Voici un exemple :

```
1 <?php
2 $firstArray = array(cle1 => element1, cle2 => element2, cle3 => element3);
3 $secondArray = [cle1 => element1, cle2 => element2, cle3 => element3];
```

Exemple

Dans cet exemple, les clés sont les lettres 'a', 'b' et 'c', et les valeurs correspondantes sont les chaînes de caractères 'PHP', 'Javascript' et 'Python' :

```
1 <?php
2
3 $languages = array("a" => "PHP", "b" => "Javascript", "c" => "Python");
```

Penchons-nous ici sur l'explication de chaque ligne de code de sorte que vous compreniez de quoi il s'agit :

- La ligne <?php est une balise d'ouverture du code PHP. Elle indique que le code qui suit sera du code PHP et sera interprété par le serveur.
- La variable \$languages est déclarée et initialisée en utilisant la fonction array(). Cette fonction crée un tableau. Les clés 'a', 'b' et 'c' sont associées respectivement aux valeurs 'PHP', 'Javascript' et 'Python'.
- Les clés du tableau sont des chaînes de caractères entourées de guillemets simples (') et suivies du symbole →, qui est l'opérateur d'association des clés et des valeurs dans un tableau associatif.
- Enfin, les valeurs du tableau sont également des chaînes de caractères entourées de guillemets simples (').

Complément

La clé peut être soit un entier (on parle alors de tableau indexé), soit une chaîne de caractère (dans ce cas, il s'agit d'un tableau associatif). La valeur peut être de n'importe quel type, y compris un tableau.

Exemple

Voyons ensemble un exemple de code PHP qui crée un tableau associatif dans la variable "\$students" :

```
1 <?php
2
3 $students = [
4   'Class1' => ['John', 'Mary', 'Karim'],
5   'Class2' => ['Jane', 'Richard', 'Anna']
6 ];
```

Le code PHP fourni crée un tableau associatif nommé \$students. Dans ce tableau, il y a deux entrées qui représentent des classes. La première entrée a pour clé « Class1 » et pour valeur un tableau contenant les noms des étudiants « John », « Mary » et « Karim ». La deuxième entrée, quant à elle, a pour clé « Class2 » et pour valeur un tableau contenant les noms des étudiants « Jane », « Richard » et « Anna ».

Ce tableau associatif permet d'organiser les étudiants par classe. Les clés du tableau représentent les noms des classes et les valeurs associées sont des tableaux contenant les noms des étudiants de chaque classe.

B. Les tableaux indexés

Les tableaux indexés sont des tableaux, dont les clés, ou index sont de type entier numérique. Chaque valeur du tableau est associée à un index. En outre, les tableaux indexés sont utilisés lorsque vous souhaitez stocker des éléments dans un ordre spécifique et accéder à ces éléments en utilisant un index numérique. Les indices commencent généralement à 0 et sont incrémentés de manière séquentielle.

Exemple

Dans le tableau \$languages suivant, la première valeur 'PHP' est associée à l'index 1, la deuxième valeur 'Javascript' est associée à l'index 2, et la troisième valeur 'Python' est associée à l'index 3. Pour accéder à une valeur spécifique dans un tableau indexé, vous devez utiliser l'index correspondant. Par exemple, pour obtenir la valeur 'Javascript' du tableau \$languages, vous utilisez \$languages[2], car 'Javascript' est associé à l'index 2.

```
1 <?php
2
3 $languages = array(1 => "PHP", 2 => "Javascript", 3 => "Python");
4 var_dump($languages) ;
5 //array(3) {
6 //   [1]=>
7 //   string(3) "PHP"
8 //   [2]=>
9 //   string(10) "Javascript"
10 //   [3]=>
11 //   string(6) "Python"
12 //}
```

Remarque

La fonction var_dump en PHP est utilisée pour afficher des informations détaillées sur une variable, y compris son type, sa valeur et sa structure. Elle est principalement utilisée à des fins de débogage et d'inspection des variables lors du développement d'une application.

Si les clés ne sont pas précisées, PHP indexe automatiquement chaque valeur du tableau, en commençant à l'index 0.

Exemple

Voici un exemple de tableau dans lequel les clés ne sont pas précisées :

```
1 <?php
2
3 $languages = array('PHP', 'Javascript', 'Python');
4 var_dump($languages) ;
5 //array(3) {
6 //   [0]=>
7 //   string(3) "PHP"
8 //   [1]=>
9 //   string(10) "Javascript"
10 //   [2]=>
11 //   string(6) "Python"
12 //}
```

Complément

Pour accéder au dernier élément d'un tableau indexé à partir de 0, il est possible d'utiliser la fonction interne `count()`. L'index du dernier élément est égal à la taille du tableau - 1.

Exemple

```
1 <?php
2
3 $languages = array('PHP', 'Javascript', 'Python');
4 $lastElement = $languages[count($languages) - 1];
5 var_dump($lastElement);
6 // string(6) "Python"
```

Dans ce code PHP, nous avons un tableau appelé `$languages` qui contient trois éléments : 'PHP', 'Javascript' et 'Python'. Ensuite, nous utilisons la fonction `count($languages)` pour obtenir le nombre total d'éléments dans le tableau `$languages`. Dans notre cas, la valeur retournée est 3. Nous utilisons cette valeur pour accéder à l'index du dernier élément du tableau en soustrayant 1, car les indices de tableau commencent à 0. Ainsi, `$languages[count($languages) - 1]` fait référence au dernier élément du tableau `$languages`, qui est 'Python'.

Enfin, nous utilisons la fonction `var_dump` pour afficher des informations détaillées sur cette variable `$lastElement`. La sortie affiche le type de la variable (string) et sa longueur (6). Ensuite, la valeur de la variable est affichée, qui est "Python" dans notre cas.

Remarque

Il est possible de ne spécifier que certains index comme dans cet exemple :

```
1 <?php
2
3 $languages = array('PHP', 2 => 'Javascript', 'Python');
4 var_dump($languages) ;
5 //array(3) {
6 //   [0]=>
7 //   string(3) "PHP"
8 //   [2]=>
9 //   string(10) "Javascript"
10 //   [3]=>
```

```
11 // string(6) "Python"
12 //}
```

C. Les tableaux associatifs

On parle de tableau associatif quand les clés sont des chaînes de caractères.

Exemple

Voici un exemple dans lequel nous utilisons le langage de programmation PHP pour définir un tableau associatif nommé « *isoCodes* » :

```
1 <?php
2 $isoCodes = [
3 'Argentine' => 'AR',
4 'Belgique' => 'BE',
5 'Chili' => 'CL',
6 'Equateur' => 'EC'
7 ];
```

Ce tableau contient des paires clé-valeur où chaque clé représente le nom d'un pays et chaque valeur correspond à son code ISO.

Il existe des règles de conversion des clés dans certains cas.

Tout d'abord, si la clé est une chaîne de caractères contenant un entier, elle sera convertie automatiquement en entier :

```
1 <?php
2 $languages = array('1' => 'PHP', 'b' => 'Javascript', 'c' => 'Python');
3 var_dump($languages) ;
4 //array(3) {
5 //   [1]=>
6 //   string(3) "PHP"
7 //   ['b']=>
8 //   string(10) "Javascript"
9 //   ['c']=>
10 //   string(6) "Python"
11 //}
```

De plus, si les nombres à virgule sont convertis à l'entier inférieur, ce qui signifie que tout ce qui est après la virgule est supprimé :

```
1 <?php
2
3 $languages = array(1.6 => 'PHP', 'b' => 'Javascript', 'c' => 'Python');
4 var_dump($languages) ;
5 //array(3) {
6 //   [1]=>
7 //   string(3) "PHP"
8 //   ['b']=>
9 //   string(10) "Javascript"
10 //   ['c']=>
11 //   string(6) "Python"
12 //}
```

Les booléens, c'est-à-dire `true` et `false`, sont convertis en entier ; `true` devient donc 1 et `false` devient 0 :

```
1 <?php
2 $languages = array(false => 'PHP', true => 'Javascript', 'Python');
3 var_dump($languages) ;
4 //array(3) {
5 //    [0]=>
6 //    string(3) "PHP"
7 //    [1]=>
8 //    string(10) "Javascript"
9 //    [2]=>
10 //    string(6) "Python"
11 //}
```

Quant à lui, `Null` devient une chaîne de caractères vide.

```
1 <?php
2 $languages = array(null => 'PHP', 'Javascript', 'Python');
3 var_dump($languages) ;
4 //array(3) {
5 //    ['']=>
6 //    string(3) "PHP"
7 //    [0]=>
8 //    string(10) "Javascript"
9 //    [1]=>
10 //    string(6) "Python"
11 //}
```

Les tableaux et objets ne peuvent pas être utilisés comme clés. Cela déclenche une erreur du type “illegal offset type” :

```
1 <?php
2 $languages = array(true => 'PHP', '1' => 'Javascript', 1.3 => 'Python');
3 var_dump($languages) ;
4 //array(1) {
5 //    [1]=>
6 //    string(6) "Python"
7 //}
```

Attention

Veillez à ne pas utiliser deux fois la même clé dans un tableau : seule la dernière est prise en compte, les autres sont écrasées.

D. Exercice : Quiz

[solution n°1 p.21]

Question 1

Un tableau s'initialise toujours vide.

- ☐ Vrai
- ☐ Faux

Question 2

De quel type de tableau s'agit-il ici ?

```
1 <?php
2
3 $myArray = [
4 '1' => 'The 4 Musketeers',
5 'a' => 'The Hitchhiker's Guide To The Galaxy',
6 'code' => 'The Hacker Ethic and the Spirit of the Information Age',
7 ];
```

- ☐ Un tableau associatif
- ☐ Un tableau indexé
- ☐ Un tableau nominatif

Question 3

Quelle valeur ce code va-t-il afficher ?

```
1 <?php
2
3 $cities = [
4 'Paris',
5 'Toulouse',
6 'Lyon',
7 'Rennes'
8 ];
9
10 echo $cities[2];
```

- ☐ 'Toulouse'
- ☐ 'Lyon'
- ☐ 'Paris'

Question 4

Quel est l'index du dernier élément du tableau ('horse') ?

```
1 <?php
2
3 $animals = [
4 'cat',
5 'dog',
6 4 => 'sheep',
7 'horse'
8 ];
```

- ☐ 3
- ☐ 4
- ☐ 5

Question 5

Combien de paires clé-valeur ce tableau contient-il ?

```
1 <?php
2
3 $animals = [
4 true => 'cat',
5 1 => 'dog',
```



```
6 '1' => 'horse',  
7 'a' => 'sheep',  
8];
```

- ☐ 2
- ☐ 4
- ☐ 3

III. Manipuler les tableaux

A. Les manipulations de base

Pour afficher un élément d'un tableau, on utilise sa clé ou son index entre crochets. Si la clé est une chaîne de caractère, il faut conserver les guillemets.

Exemple

```
1 <?php  
2  
3 $countries = ['Argentine', 'Belgique', 'Chili', 'Equateur'];  
4 $isoCode = [  
5 'Argentine' => 'AR',  
6 'Belgique' => 'BE',  
7 'Chili' => 'CL',  
8 'Equateur' => 'EC'  
9];  
10 echo $countries[1]; // Affiche Belgique'  
11 echo $isoCode['Argentine']; // Affiche 'AR'
```

Pour afficher tous les éléments d'un tableau, on peut utiliser une boucle `foreach`. Cette boucle passe en revue chaque élément du tableau passé en argument et exécute le code spécifié sur chaque paire clé-valeur, selon la syntaxe suivante : `foreach($array as $key => value) { commande }`.

Exemple

```
1 <?php  
2  
3 $names = ['Lea', 'Morgan', 'Lionel', 'Marina'];  
4 foreach($names as $key => $name) {  
5     echo "Le prénom " . $name . " est à la clé " . $key . " du tableau.\n";  
6 };  
7  
8 //Le prénom Lea est à la clé 0 du tableau.  
9 //Le prénom Morgan est à la clé 1 du tableau.  
10 //Le prénom Lionel est à la clé 2 du tableau.  
11 //Le prénom Marina est à la clé 3 du tableau.
```

Complément

Si un seul argument est passé à la boucle `foreach`, elle ne passe en revue que les valeurs, et ignore les clés du tableau.

Exemple

```
1 <?php
2
3 $names = ['Lea', 'Morgan', 'Lionel', 'Marina'];
4 foreach($names as $name) {
5     echo $name . "\n";
6 };
7
8 //Lea
9 //Morgan
10 //Lionel
11 //Marina
```

Pour accéder à la valeur d'un tableau multidimensionnel, il faut en préciser les clés successivement entre crochets.

Exemple

```
1 <?php
2
3 $countries = [
4     'Europe' => ['France', 'Belgium', 'Germany'],
5     'America' => ['Brazil', 'United States', 'Mexico'],
6     'Asia' => ['India', 'China'],
7     'Africa' => ['Senegal', 'Mali']
8 ];
9 echo $countries['Asia'][0]; // 'India'
```

`$countries['Asia'][0]` renvoie au premier élément du tableau qui a la clé 'Asia', dans le tableau `$countries`.

Il est possible de modifier la valeur d'un élément d'un tableau, c'est-à-dire d'assigner à la clé une nouvelle valeur. Par exemple, si l'élément situé à l'index 0 de votre tableau `$array` n'est pas correctement orthographié, vous pouvez assigner une nouvelle valeur en accédant à `$array[0]` suivi du signe = et de la nouvelle valeur.

Exemple

```
1 <?php
2
3 $countries = [
4     'Europe' => ['France', 'Belgium', 'Germany'],
5     'America' => ['Brazil', 'United States', 'Mexico'],
6     'Asia' => ['India', 'China'],
7     'Africa' => ['Senegal', 'Mali']
8 ];
9 $countries['Asia'][0] = 'Japan';
10 echo $countries['Asia'][0]; // 'Japan'
```

Pour ajouter des éléments à un tableau, il existe plusieurs méthodes. La première consiste à empiler l'élément à la fin du tableau.

Exemple

```
1 <?php
2
3 $names = ['Lea', 'Morgan', 'Lionel', 'Marina'];
4 $names[] = 'Manon';
5 var_dump($names);
6 //array(5) {
7 // [0]=>
8 //string(3) "Lea"
9 // [1]=>
10 //string(6) "Morgan"
11 // [2]=>
12 //string(6) "Lionel"
13 // [3]=>
14 //string(6) "Marina"
15 // [4]=>
16 //string(5) "Manon"
```

Il est également possible d'ajouter un élément en précisant son index.

Exemple

```
1 <?php
2
3 $names = ['Lea', 'Morgan', 'Lionel', 'Marina'];
4 $names[4] = 'Manon';
5 var_dump($names);
6 //array(5) {
7 // [0]=>
8 //string(3) "Lea"
9 // [1]=>
10 //string(6) "Morgan"
11 // [2]=>
12 //string(6) "Lionel"
13 // [3]=>
14 //string(6) "Marina"
15 // [4]=>
16 //string(5) "Manon"
```

B. Les fonctions système des tableaux

Il existe un certain nombre de fonctions internes à PHP permettant de travailler sur les tableaux.

La documentation officielle de PHP les liste toutes : [php¹](https://www.php.net/manual/fr/ref.array.php).

Voilà quelques-unes des plus utiles :

- `count($array)` : compte le nombre d'éléments dans un tableau. Valeur de retour : un entier.
- `in_array($needle, $haystack, $strict = false)` : vérifie si un élément existe dans un tableau donné. Si `$strict` n'est pas passé à `true`, la comparaison ne se fait pas sur le type de valeur. Valeur de retour : un booléen.
- `is_array($array)` : vérifie si une valeur est de type tableau. Valeur de retour : un booléen. Cette fonction est utile pour vérifier que la valeur reçue est bien du type attendu.

1 <https://www.php.net/manual/fr/ref.array.php>

- `array_unique($array, int $flags = SORT_STRING)` : supprime les valeurs en doublon d'un tableau. Le deuxième argument, optionnel, permet de préciser le comportement de comparaison. Valeur de retour : un nouveau tableau (le tableau d'origine n'est pas modifié).
- `array_reverse($array, $preserve_keys = false)` : retourne un tableau dont les éléments sont en ordre inversé par rapport au tableau d'origine. Les clés en chaîne de caractères ne sont pas affectées, mais les clés sous forme d'entier sont modifiées. Si le deuxième argument est à `true`, les clés numériques seront préservées. Valeur de retour: un tableau.
- `sort(&$array, $flags = SORT_REGULAR)` : trie un tableau sur place (le tableau d'origine est modifié) en ordre croissant. Le deuxième paramètre est optionnel et permet de modifier le comportement de tri. Valeur de retour: `true`.
- `rsort(&$array, $flags = SORT_REGULAR)` : trie un tableau sur place (le tableau d'origine est modifié) en ordre décroissant. Le deuxième paramètre est optionnel et permet de modifier le comportement de tri. Valeur de retour: `true`.
- `implode($separator, $array)` : transforme un tableau en chaîne de caractères. Chaque valeur du tableau est séparée de la suivante par la chaîne de caractères passée en premier argument. Valeur de retour: une chaîne de caractères.
- `max($array)` : renvoie la valeur la plus élevée d'un tableau.
- `min($array)` : renvoie la plus petite valeur d'un tableau.

Exemple

Prenons l'exemple d'un programme qui reçoit une liste de films depuis une base de données, et doit les afficher à l'utilisateur en prenant soin d'abord de supprimer d'éventuels doublons, puis en les triant afin de les afficher par ordre alphabétique. Puisque les fonctions qui permettent de supprimer les doublons et de trier les éléments ne s'appliquent que sur des tableaux, il est utile de vérifier d'abord que les données reçues sont bien du type tableau, afin d'éviter toute erreur. Finalement, les films doivent être transformés en chaîne de caractères, séparés par une virgule et un espace.

```
1 <?php
2
3 $films = "Princess Mononoke", 'Matrix', 'Children of Men', 'Matrix', 'Moon';
4
5 function unifyAndSortToString($list) {
6     if (!is_array($list)) {
7         return;
8     };
9     $listUnified = array_unique($list);
10    sort($listUnified);
11    return implode(', ', $listUnified);
12 };
13
14 //string(48) "Children of Men, Matrix, Moon, Princess Mononoke"
```

- `array_map($callback, $array)` : applique une fonction de rappel sur tous les éléments d'un tableau, sans modifier le tableau d'origine. Valeur de retour: un nouveau tableau.
- `array_filter($array, $callback)` : applique un filtre à tous les éléments d'un tableau selon une fonction de rappel. Valeur de retour: un nouveau tableau.

Complément

Enfin, il existe certaines fonctions utiles spécifiquement pour les tableaux associatifs :

- `array_key_exists($key, $array)` : vérifie si une clé existe dans un tableau. Valeur de retour: un booléen.
- `array_keys($array)` : retourne toutes les clés d'un tableau. Valeur de retour : un nouveau tableau.
- `array_values($array)` : retourne toutes les valeurs d'un tableau. Valeur de retour: un nouveau tableau.
- `asort(&$array, $flags = SORT_REGULAR)` : trie les valeurs d'un tableau sur place en ordre croissant, en conservant l'association clé-valeur. Le deuxième paramètre est optionnel, il permet de modifier le comportement de tri. Le tableau d'origine est modifié. Valeur de retour: `true`.
- `arsort(&$array, $flags = SORT_REGULAR)` : trie les valeurs d'un tableau en ordre décroissant, en conservant l'association clé-valeur. Le deuxième paramètre est optionnel, il permet de modifier le comportement de tri. Le tableau d'origine est modifié. Valeur de retour: `true`.
- `ksort(&$array, $flags = SORT_REGULAR)` : trie les clés d'un tableau sur place en ordre croissant, en conservant l'association clé-valeur. Le deuxième paramètre est optionnel, il permet de modifier le comportement de tri. Le tableau d'origine est modifié. Valeur de retour: `true`.
- `krsort(&$array, $flags = SORT_REGULAR)` : trie les clés d'un tableau sur place en ordre décroissant, en conservant l'association clé-valeur. Le deuxième paramètre est optionnel, il permet de modifier le comportement de tri. Le tableau d'origine est modifié. Valeur de retour: `true`.

Exemple

Prenons une liste des plus hauts sommets du monde, où chaque clé correspond à un continent, et la valeur à un sommet qui lui est associé. Imaginons que vous ayez d'abord à vérifier qu'il contient bien une clé correspondant au continent 'Asia', puis à l'afficher par ordre croissant de clés.

```
1 <?php
2
3 $mountains = array('Asia' => 'Everest', 'Africa' => 'Kilimandjaro', 'America' => 'Denali');
4 if (array_key_exists('Asia', $mountains)) ksort($mountains);
5 var_dump($mountains);
6
7 //array(3) {
8 //["Africa"]=>
9 //string(12) "Kilimandjaro"
10 //["America"]=>
11 //string(6) "Denali"
12 //["Asia"]=>
13 //string(7) "Everest"
14 //}
```

C. Exercice : Quiz

[solution n°2 p.22]

Question 1

Comment accéder à la valeur 'K2' ?

```
1 <?php
2
3 $mountains = array('Asia' => ['Everest', 'K2'], 'Africa' => ['Kilimandjaro'], 'America' =>
  ['Denali']);
```

- ☐ \$mountains[3]
- ☐ \$mountains[1][2]
- ☐ \$mountains['Asia'][1]

Question 2

Comment remplacer la valeur 'United-States' par 'Canada' dans le tableau \$countries ?

```
1 <?php
2
3 $countries = [
4   'Europe' => ['France', 'Belgium', 'Germany'],
5   'America' => ['Brazil', 'United-States', 'Mexico'],
6   'Asia' => ['India', 'China'],
7   'Africa' => ['Senegal', 'Mali']
8];
```

- ☐ \$countries['America'][1] = 'Canada';
- ☐ \$countries['Armerica'][] = 'Canada';
- ☐ \$countries[2][1] = 'Canada';

Question 3

Quelle sera la valeur de \$fruits à l'exécution du code suivant ?

```
1 <?php
2 $fruits = ['banana', 'ananas', 'pear', 'apple'];
3 sort($fruits);
```

- ☐ ['banana', 'ananas', 'pear', 'apple']
- ☐ ['pear', 'banana', 'apple', 'ananas']
- ☐ ['ananas', 'apple', 'banana', 'pear']

Question 4

Quelle fonction permet de parcourir les éléments d'un tableau en leur appliquant une fonction de retour ?

- ☐ array_map()
- ☐ array_search()
- ☐ is_array()

Question 5

Quelle sera la valeur de \$students après l'exécution du code suivant ?

```
1 <?php
2
3 $students = ['marine', 'julien', 'nicolas', 'assia'];
4 function toUpperCase($name){
5     return strtoupper($name);
6 };
7 $studentsToUpperCase = array_map('toUpperCase', $students);
```

- ☐ ['marine', 'julien', 'nicolas', 'assia']
- ☐ ['MARINE', 'JULIEN', 'NICOLAS', 'ASSIA']
- ☐ ['assia', 'julien', 'marine', 'nicolas']

IV. Les opérateurs de tableau

A. Introduction

Les opérateurs permettent de retourner une nouvelle valeur à partir d'une ou plusieurs expressions. Il en existe plusieurs types, voir la documentation officielle de PHP pour une liste exhaustive : [php¹](https://www.php.net/manual/fr/language.operators.php). Dans le cas des tableaux, on utilise les opérateurs de comparaison, ainsi qu'un seul opérateur arithmétique.

B. Les opérateurs de comparaison

Les opérateurs de comparaison permettent de comparer deux tableaux et renvoient un booléen.

Égalité : `==`

L'opérateur d'égalité `==` renvoie `true` si deux tableaux contiennent les mêmes paires clés-valeurs. L'opérateur ne prend pas en compte l'ordre des valeurs ni leur type.

Exemple

```
1 <?php
2
3 $listA = ['a' => 'PHP', 'b' => 'Javascript', 'c' => 'Python', 'd' => 3];
4 $listB = ['b' => 'Javascript', 'a' => 'PHP', 'c' => 'Python', 'd' => '3'];
5 var_dump($listA == $listB); // true
```

L'opérateur de Stricte égalité `===` renvoie `true` si deux tableaux contiennent les mêmes paires clés-valeurs, du même type, et dans le même ordre.

Exemple

```
1 <?php
2
3 $listA = ['a' => 'PHP', 'b' => 'Javascript', 'c' => 'Python', 'd' => 3];
4 $listB = ['b' => 'Javascript', 'a' => 'PHP', 'c' => 'Python', 'd' => '3'];
5 var_dump($listA === $listB); // false
```

Les deux opérateurs d'inégalité `!=` ou `<>` ont le même fonctionnement. Ils renvoient `true` si deux tableaux n'ont pas les mêmes paires clé-valeur. Ils ne prennent pas en compte le type et l'ordre des valeurs.

1 <https://www.php.net/manual/fr/language.operators.php>

Exemple

```
1 <?php
2
3 $listA = ['a' => 'PHP', 'b' => 'Javascript', 'c' => 'Python', 'd' => 3];
4 $listB = ['b' => 'Javascript', 'a' => 'PHP', 'c' => 'Python'];
5 var_dump($listA != $listB); // true
```

L'opérateur de stricte inégalité `!==` renvoie `true` si deux tableaux ne sont pas strictement identiques, c'est-à-dire s'ils n'ont pas les mêmes paires clé-valeur, du même type et dans le même ordre.

Exemple

```
1 <?php
2
3 $listA = ['a' => 'PHP', 'b' => 'Javascript', 'c' => 'Python'];
4 $listB = ['b' => 'Javascript', 'a' => 'PHP', 'c' => 'Python'];
5 var_dump($listA !== $listB); // renvoie true
```

C. Opérateur arithmétique

Un seul opérateur arithmétique peut être utilisé pour travailler avec les tableaux, c'est l'opérateur d'addition : `+`. Il permet d'ajouter les valeurs du tableau à droite de l'opérateur à celles du tableau à gauche. Si une ou plusieurs clés sont présentes dans les deux tableaux, ce sont celles du tableau de gauche qui sont conservées, celles du tableau de droite sont ignorées.

D. Exercice : Quiz

[solution n°3 p.24]

Question 1

Quel opérateur permet de vérifier l'égalité entre deux tableaux, en prenant en compte le type et l'ordre des paires clés-valeurs ?

- ☐ `==`
- ☐ `<>`
- ☐ `===`

Question 2

L'opérateur d'inégalité (`!=` ou `<>`) renvoie `true` si deux tableaux n'ont pas les mêmes paires clés-valeurs.

- ☐ Vrai
- ☐ Faux

Question 3

Quelle sera la valeur de `$boolean` à l'exécution du code suivant ?

```
1 <?php
2
3 $arrayA = ['a' => 'apple', 2 => 'banana', '3' => 'cherry'];
4 $arrayB = ['a' => 'apple', 3 => 'cherry', '2' => 'banana'];
5 $boolean = $arrayA == $arrayB;
```


- ☐ True
- ☐ False
- ☐ Cela renvoie "Error value"

Question 4

L'opérateur de soustraction (-) permet de soustraire des éléments d'un tableau.

- ☐ Vrai
- ☐ Faux

Question 5

Quelle sera la valeur de \$isoCodes à l'exécution du code suivant ?

```
1 <?php
2
3 $isoCodeA = [
4     'Argentine' => 'AR',
5     'Equateur' => 'EC'
6 ];
7 $isoCodeB = [
8     'Equateur' => 'EC',
9     'Japon' => 'JP'
10 ];
11 $isoCodes = $isoCodeB + $isoCodeA;
```

- ☐ ['Argentine' => 'AR', 'Equateur' => 'EC', 'Japon' => 'JP'];
- ☐ ['Equateur' => 'EC', 'Japon' => 'JP', 'Argentine' => 'AR'];
- ☐ ['Argentine' => 'AR', 'Equateur' => 'EC', 'Equateur' => 'EC', 'Japon' => 'JP'];

V. Essentiel

Les tableaux sont un type de données extrêmement utilisé. Ils permettent de stocker et de manipuler de grandes quantités de données, facilement accessibles grâce à leurs clés, qu'elles soient numériques (tableaux indexés) ou en chaîne de caractère (tableaux associatifs). La boucle foreach permet d'itérer sur toutes les paires clés-valeurs d'un tableau.

Les fonctions internes à PHP permettent de pratiquer un grand nombre d'opérations sur les tableaux. Il est important de garder à l'esprit que certaines de ces fonctions vont modifier le tableau d'origine (les fonctions de tri, par exemple) alors que d'autres non. Certaines fonctions peuvent appliquer une fonction de rappel sur chaque élément d'un tableau. Elles peuvent être utiles pour modifier tous les éléments du tableau, ou pour opérer un tri et ne conserver que ceux qui remplissent une condition prédéfinie.

Certains opérateurs s'appliquent sur les tableaux : il s'agit des opérateurs de comparaison (égalité, égalité stricte, inégalité, inégalité stricte) et de l'opérateur d'addition.

VI. Auto-évaluation

A. Exercice

Vous développez une application de jeux pédagogiques pour les enfants. Vous disposez d'une première base de données sous forme de tableau comprenant une liste de noms, à laquelle est associé l'âge des enfants. Vous recevez un nouveau tableau avec de nouveaux utilisateurs.

```
1 $users = [
2     'Adam' => 8,
3     'Julie' => 13,
4     'Karima' => 11,
5     'Anna' => 11,
6     'Marina' => 9,
7     'Mohamed' => 7,
8     'Arthur' => 12,
9     'Morgan' => 14
10 ];
11
12 $newUsers = [
13     'Hector' => 6,
14     'Manon' => 8,
15     'Elisa' => 10,
16     'Leo' => 12,
17     'Enzo' => 13,
18     'Ada' => 9
19 ];
```

Question 1

[solution n°4 p.25]

Comment ajouter les données du deuxième tableau (\$newUsers) au premier (\$users), en évitant d'éventuels doublons ?

Question 2

[solution n°5 p.25]

Comment trier les valeurs du tableau par ordre d'âge croissant, puis les filtrer pour créer deux nouveaux tableaux, un contenant tous les enfants de moins de 10 ans, l'autre tous les enfants de 10 ans et plus ?

B. Test

Exercice 1 : Quiz

[solution n°6 p.26]

Question 1

La boucle foreach est le seul moyen d'itérer sur les éléments d'un tableau.

- ☐ Vrai
- ☐ Faux

Question 2

Que renvoie le code suivant ?

```
1 <?php
2 $isOver18 = true;
3 var_dump(is_array($isOver18));
```

- ☐ True
- ☐ False
- ☐ Cela renvoie "Error var_dump"

Question 3

Quelle est la valeur de l'élément situé à l'index \$students['secondYear']['classB'][3] ?

```
1 <?php
2 $students = [
3     'firstYear' => [
4         'classA' => ['Nathalie', 'Anne', 'Arthur', 'Steve'],
5         'classB' => ['Amine', 'Henri', 'Pierre', 'Alix']
6     ],
7     'secondYear' => [
8         'classA' => ['Pauline', 'Florian', 'Cécile', 'Emilie'],
9         'classB' => ['Oriane', 'Nicolas', 'Olivier', 'Laura']
10    ],
11    'thirdYear' => [
12        'classA' => ['Rémi', 'Julie', 'Mélodie', 'Marion'],
13        'classB' => ['Marina', 'Léa', 'Lionel', 'Manu']
14    ]
15 ]
```

- ☐ Olivier
- ☐ Laura
- ☐ Marion

Question 4

Parmi les fonctions suivantes, quelle fonction de rappel utiliser avec array_filter pour ne garder que les éléments du tableau \$ages supérieurs ou égaux à 18 ?

```
1 $ages = [23, 5, 17, 45, 87, 54, 4, 24, 12];
2 function filterA($age) {
3     return $age >= 18;
4 };
5 function filterB($age) {
6     return $age + 18;
7 }
8 function filterC($age) {
9     return $age === 18;
10 }
```

- ☐ filterA
- ☐ filterB
- ☐ filterC

Question 5

Quelle est la valeur de \$boolean à l'exécution du code suivant ?

```
1 $groupA = [
2     '1' => 'elephant',
3     2 => 'mouse',
4     3 => 'pig'
5 ];
6 $groupB = [
7     1 => 'elephant',
8     3 => 'pig',
9     '2' => 'mouse'
10 ];
11 $boolean = $groupA == $groupB;
```

- ☐ True
- ☐ False
- ☐ Null


Solutions des exercices

Exercice p. 7 Solution n°1**Question 1**

Un tableau s'initialise toujours vide.

☐ Vrai

☒ Faux

 Un tableau peut être initialisé vide ou bien avec un nombre illimité de valeurs.

Question 2


De quel type de tableau s'agit-il ici ?

```
1 <?php
2
3 $myArray = [
4 '1' => 'The 4 Musketeers',
5 'a' => 'The Hitchhiker's Guide To The Galaxy',
6 'code' => 'The Hacker Ethic and the Spirit of the Information Age',
7 ];
```

☒ Un tableau associatif

☐ Un tableau indexé

☐ Un tableau nominatif

 Il s'agit d'un tableau associatif, puisque ses clés sont de type chaîne de caractères.

Question 3


Quelle valeur ce code va-t-il afficher ?

```
1 <?php
2
3 $cities = [
4 'Paris',
5 'Toulouse',
6 'Lyon',
7 'Rennes'
8 ];
9
10 echo $cities[2];
```

☐ 'Toulouse'

☒ 'Lyon'

☐ 'Paris'

 L'index des éléments d'un tableau commençant à 0, l'élément situé à l'index 2 est le troisième élément du tableau.

Question 4


Quel est l'index du dernier élément du tableau ('horse') ?

```
1 <?php
2
3 $animals = [
4 'cat',
5 'dog',
6 4 => 'sheep',
7 'horse'
8 ];
```

☐ 3

☐ 4

☒ 5

 L'élément précédent étant indexé à 4, la valeur 'horse' se voit affecter automatiquement l'index 5.

Question 5


Combien de paires clé-valeur ce tableau contient-il ?

```
1 <?php
2
3 $animals = [
4 true => 'cat',
5 1 => 'dog',
6 '1' => 'horse',
7 'a' => 'sheep',
8 ];
```

☒ 2

☐ 4

☐ 3

 Trois éléments du tableau ont la même clé : true, 1 et '1', toutes converties en entier 1. Seule la dernière paire clé-valeur est conservée, les deux autres sont écrasées. Le tableau contient donc deux paires clés-valeurs.

Exercice p. 13 Solution n°2

Question 1


Comment accéder à la valeur 'K2' ?

```
1 <?php
2
3 $mountains = array('Asia' => ['Everest', 'K2'], 'Africa' => ['Kilimandjaro'], 'America' =>
  ['Denali']);
```

☐ \$mountains[3]

☐ \$mountains[1][2]

☒ \$mountains['Asia'][1]


 \$mountains est un tableau multidimensionnel. La valeur 'K2' est dans le tableau qui a la clé 'Asia', à l'index 1.

Question 2

Comment remplacer la valeur 'United-States' par 'Canada' dans le tableau \$countries ?

```
1 <?php
2
3 $countries = [
4   'Europe' => ['France', 'Belgium', 'Germany'],
5   'America' => ['Brazil', 'United-States', 'Mexico'],
6   'Asia' => ['India', 'China'],
7   'Africa' => ['Senegal', 'Mali']
8 ];
```

- ☒ \$countries['America'][1] = 'Canada';
- ☐ \$countries['Armerica'][] = 'Canada';
- ☐ \$countries[2][1] = 'Canada';


 La valeur 'United-States' se trouve dans le tableau qui est à la clé 'America', à l'index 1. La deuxième syntaxe ajoute la valeur 'Canada' à la fin du tableau qui est à la clé 'America'. La troisième ajoute une nouvelle paire clé valeur dans le tableau, 2 => 'Canada'.

Question 3

Quelle sera la valeur de \$fruits à l'exécution du code suivant ?

```
1 <?php
2 $fruits = ['banana', 'ananas', 'pear', 'apple'];
3 sort($fruits);
```


- ☐ ['banana', 'ananas', 'pear', 'apple']
- ☐ ['pear', 'banana', 'apple', 'ananas']
- ☒ ['ananas', 'apple', 'banana', 'pear']

 La fonction sort() trie un tableau sur place, par ordre croissant. Le tableau d'origine, \$fruits, est donc modifié.

Question 4

Quelle fonction permet de parcourir les éléments d'un tableau en leur appliquant une fonction de retour ?

- ☒ array_map()
- ☐ array_search()
- ☐ is_array()

 array_map applique une fonction de retour à chaque élément d'un tableau. La fonction de retour est passée en premier argument. array_search permet de trouver la première clé associée à la valeur recherchée. is_array permet de savoir si une variable est du type tableau.

Question 5


Quelle sera la valeur de \$students après l'exécution du code suivant ?

```
1 <?php
2
3 $students = ['marine', 'julien', 'nicolas', 'assia'];
4 function toUpperCase($name){
5     return strtoupper($name);
6 };
7 $studentsToUpperCase = array_map('toUpperCase', $students);
```

☒ ['marine', 'julien', 'nicolas', 'assia']

☐ ['MARINE', 'JULIEN', 'NICOLAS', 'ASSIA']

☐ ['assia', 'julien', 'marine', 'nicolas']

 array_map ne modifie pas le tableau d'origine. À la fin de l'exécution du code, le tableau \$students n'a pas été changé. Les changements opérés par array_map ont été enregistrés dans une nouvelle variable, \$studentsToUpperCase.

Exercice p. 16 Solution n°3


Question 1

Quel opérateur permet de vérifier l'égalité entre deux tableaux, en prenant en compte le type et l'ordre des paires clés-valeurs ?

☐ ==

☐ <>

☒ ===


 L'opérateur === renvoie true si deux tableaux sont strictement identiques.

Question 2

L'opérateur d'inégalité (!= ou <>) renvoie true si deux tableaux n'ont pas les mêmes paires clés-valeurs.

☒ Vrai

☐ Faux

 L'opérateur != renvoie true si deux tableaux n'ont pas les mêmes paires clés-valeurs.

Question 3

Quelle sera la valeur de \$boolean à l'exécution du code suivant ?

```
1 <?php
2
3 $arrayA = ['a' => 'apple', 2 => 'banana', '3' => 'cherry'];
4 $arrayB = ['a' => 'apple', 3 => 'cherry', '2' => 'banana'];
5 $boolean = $arrayA == $arrayB;
```

☒ True

☐ False

☐ Cela renvoie "Error value"

- Q L'opérateur == renvoie true si deux tableaux sont identiques, sans tenir compte de l'ordre des paires clés-valeurs ni du type des clés.

Question 4

L'opérateur de soustraction (-) permet de soustraire des éléments d'un tableau.

☐ Vrai

☒ Faux

- Q Les seuls opérateurs qui s'appliquent à la manipulation des tableaux sont les opérateurs de comparaison, et l'opérateur d'addition (+).

Question 5

Quelle sera la valeur de \$isoCodes à l'exécution du code suivant ?

```
1 <?php
2
3 $isoCodeA = [
4     'Argentine' => 'AR',
5     'Equateur' => 'EC'
6 ];
7 $isoCodeB = [
8     'Equateur' => 'EC',
9     'Japon' => 'JP'
10 ];
11 $isoCodes = $isoCodeB + $isoCodeA;
```

☐ ['Argentine' => 'AR', 'Equateur' => 'EC', 'Japon' => 'JP'];

☒ ['Equateur' => 'EC', 'Japon' => 'JP', 'Argentine' => 'AR'];

☐ ['Argentine' => 'AR', 'Equateur' => 'EC', 'Equateur' => 'EC', 'Japon' => 'JP'];

- Q Le tableau à gauche de l'opérateur étant le tableau \$isoCodeB, ce sont ses valeurs qui sont conservées en premier. La valeur 'Equateur' étant présente dans les deux tableaux, elle est écrasée.

p. 18 Solution n°4

Voici comment ajouter les données du deuxième tableau (\$newUsers) au premier (\$users), en évitant d'éventuels doublons :

```
1 <?php
2 // option 1
3 $users = $users + $newUsers;
4 // option 2
5 $users += $newUsers;
```

L'opérateur d'addition permet d'ajouter les éléments d'un tableau à un autre, en supprimant du deuxième tableau (celui situé à droite de l'opérateur) les valeurs en doublon. Les deux syntaxes sont équivalentes.

p. 18 Solution n°5

Voici comment répondre à notre problématique :

```
1 <?php
2 asort($users);
3 function filterUnderTen ($age) {
4     return $age < 10;
5 };
6 function filterTenAndOver ($age) {
7     return $age >= 10;
8 };
9 $underTen = array_filter($users, 'filterUnderTen');
10 $tenAndOver = array_filter($users, 'filterTenAndOver');
```

La fonction asort() de PHP permet de trier les valeurs d'un tableau associatif en ordre croissant, tout en conservant l'association des index. Le tableau \$users est alors modifié. La fonction array_filter() itère sur les éléments d'un tableau et les filtre grâce à une fonction de rappel. Il est donc nécessaire de créer deux fonctions de rappel, l'une retournant true si la valeur est inférieure à dix, l'autre retournant true si la valeur est supérieure ou égale à 10.


Exercice p. 18 Solution n°6

Question 1

La boucle foreach est le seul moyen d'itérer sur les éléments d'un tableau.

☐ Vrai

☒ Faux

 Il existe plusieurs moyens d'itérer sur les éléments d'un tableau, parmi lesquels la boucle foreach et la fonction array_map().

Question 2


Que renvoie le code suivant ?

```
1 <?php
2 $isOver18 = true;
3 var_dump(is_array($isOver18));
```

☐ True

☒ False

☐ Cela renvoie "Error var_dump"

 La fonction is_array() renvoie true si la variable passée en argument est un tableau, false dans le cas contraire.

Question 3

Quelle est la valeur de l'élément situé à l'index \$students['secondYear']['classB'][3] ?

```
1 <?php
2 $students = [
3     'firstYear' => [
4         'classA' => ['Nathalie', 'Anne', 'Arthur', 'Steve'],
5         'classB' => ['Amine', 'Henri', 'Pierre', 'Alix']
6     ],
7     'secondYear' => [
8         'classA' => ['Pauline', 'Florian', 'Cécile', 'Emilie'],
9         'classB' => ['Oriane', 'Nicolas', 'Olivier', 'Laura']
10    ]
11];
```

```

10 ],
11 'thirdYear' => [
12   'classA' => ['Rémi', 'Julie', 'Mélodie', 'Marion'],
13   'classB' => ['Marina', 'Léa', 'Lionel', 'Manu']
14 ]

```

- ☐ Olivier
- ☒ Laura
- ☐ Marion

Q Pour accéder à la valeur d'un tableau multidimensionnel, il faut en préciser toutes les clés successivement entre crochets. L'élément 'Laura' est à l'index 3 (4^e élément) du tableau qui a pour clé 'classB', lui-même un élément du tableau qui a pour clé 'secondYear' dans le tableau \$students.

Question 4

Parmi les fonctions suivantes, quelle fonction de rappel utiliser avec `array_filter` pour ne garder que les éléments du tableau \$ages supérieurs ou égaux à 18 ?

```

1 $ages = [23, 5, 17, 45, 87, 54, 4, 24, 12];
2 function filterA($age) {
3   return $age >= 18;
4 };
5 function filterB($age) {
6   return $age + 18;
7 }
8 function filterC($age) {
9   return $age === 18;
10 }

```

- ☒ filterA
- ☐ filterB
- ☐ filterC

Q La fonction de rappel passée en argument de `array_filter` doit retourner un booléen. C'est l'expression `$age >= 18` qui permet d'évaluer si chaque élément du tableau est bien supérieur ou égal à 18.

Question 5

Quelle est la valeur de \$boolean à l'exécution du code suivant ?

```

1 $groupA = [
2   '1' => 'elephant',
3   2 => 'mouse',
4   3 => 'pig'
5 ];
6 $groupB = [
7   1 => 'elephant',
8   3 => 'pig',
9   '2' => 'mouse'
10 ];
11 $boolean = $groupA == $groupB;

```

- ☒ True
- ☐ False
- ☐ Null

- Q L'opérateur d'égalité compare les paires clés-valeurs de deux tableaux sans tenir compte de l'ordre des paires ni de leur type.