IV) Les tableaux

PHP - Initiation

Les tableaux

IV.1) Déclarer un tableau

Les tableaux sont des genres de supers variables qui permettent de stocker plusieurs données. (Les fonctions retournant plusieurs valeurs le font généralement sous la forme de tableaux). Ils peuvent être une alternative aux bases de données si les informations à manipuler ne sont pas trop nombreuses et si le stockage à long terme de ces dernières ne nous importe pas.

Il existe plusieurs façons de déclarer un tableau :

```
// méthode classique, un peu désuette :
$tableau = array();

// méthode plus courte et moderne, à privilégier :
$tableau = []
```

PHP - Initiation

Les tableaux

IV.2) Les tableaux indicés (ou scalaires)

Le premier type de tableau et le plus simple est celui des **tableaux indicés**.

Chaque élément a un indice entier positif. **Le premier élément porte l'indice zéro, ATTENTION c'est très important !** Un tableau peut se remplir en adressant chaque élément ou en fournissant les valeurs à la fonction **array()**, ou directement entre **[]**.

```
$tab[0] = 'P';
$tab[1] = 'H';
$tab[2] = 'P';

// est équivalent à :
$tab = array('P', 'H', 'P');
// également équivalent à :
$tab = ['P', 'H', 'P'];
```

PHP - Initiation

Les tableaux

IV.3) Les tableaux associatifs

Un autre type de tableau est le **tableau associatif**.
Chaque élément a une valeur associée qui sert de clé d'accès.
Ce type de tableau permet de réaliser facilement des associations type dictionnaire permettant d'accéder rapidement à un élément. On utilise la combinaison de symbole **=>** pour assigner la correspondance entre la **clé** et sa **valeur**.
Un tableau peut se remplir en adressant chaque élément ou en fournissant les valeurs à la fonction **array()**, ou directement entre **[]**.

```
$ventes['lundi'] = 1234;
$ventes['mardi'] = 5678;
$ventes['mercredi'] = 4444;

// est équivalent à :
$ventes = array('lundi' =>1234, 'mardi' =>5678, 'mercredi' =>4444);
// également équivalent à :
$ventes = ['lundi' =>1234, 'mardi' =>5678, 'mercredi' =>4444];
```

PHP - Initiation

Les tableaux

IV.4) Les tableaux multi-dimensionnels

Un **tableau multi-dimensionnel** est assez simple à comprendre si l'on considère que chaque élément d'un tableau peut lui-même être un tableau. Si l'on s'en tient à 2 dimensions, on pourrait donc imaginer que les indices du tableau principal sont les lignes et les indices des tableaux contenus sont les colonnes d'un tableau excel.

```
$lundi = ['6', '12', '21'];
$mardi = ['8', '15', '19'];
$mercredi = ['7', '21', '20'];

$semaine = [$lundis, $mardis, $mercredis];
```

En combinant les indices (ou clés) des différents tableaux, on peux facilement retrouver une donnée précise.

PHP - Initiation

Les tableaux

IV.5) Consulter une valeur

***tableau*[*indice*]**
Renvoi la **valeur** présente à l'indice ***indice***. (tableau indicé)

***tableau*[*cle*]**
Renvoi la **valeur** présente à la clé ***cle***. (tableau associatif)

***tableau*[*cle*][*indice*]**
Renvoi la **valeur** présente à l'indice ***indice*** du tableau contenu à la clé ***cle***. (tableau multi-dimensionnel)

Il est tout à fait possible de combiner des tableaux de plusieurs types.

```
$tab1 = [1,2,3,4];
$tab2 = [5,6,7,8];
$tableau = ['lundi' => $tab1, 'mardi' => $tab2];
echo $tableau['mardi'][2];      // va afficher : 7
```

PHP - Initiation

Les tableaux

IV.6) Manipulations

IV.6.1) Ajouter/ modifier des données

tableau*[] = *valeur

Ajoute ***valeur*** en fin de tableau. L'indice est crée automatiquement. (la valeur d'indice la plus grande est prise en compte pour calculer la suivante, ce qui veut dire que les éventuels 'trous' ne sont pas gérés). Cette méthode sur un tableau associatif aura pour effet de crée un indice selon les même règles. Notre tableau aura donc au final des données accessibles par des **indices** et par des **clés**.

```
$tab[0] = 1;
$tab[3] = 4;
$tab[] = 2;
print_r($tab);      // va afficher : Array ( [0] => 1 [3] => 4 [4] => 2 )
```

PHP - Initiation

Les tableaux

tableau*[*indice*] = *valeur

Associe ***valeur*** à la position ***indice***. Si une valeur existe déjà à cet indice, elle est remplacée, sinon crée.

tableau*[*cle*] = *valeur

Associe ***valeur*** à la clé ***cle***. Si une valeur existe déjà à cet clé, elle est remplacée, sinon crée.

```
$ventes = ['lundi' =>1234, 'mardi' =>5678, 'mercredi' =>4444];

$ventes['mardi'] = 2222;
$ventes['jeudi'] = 6666;

print_r($ventes);

// va afficher : Array ( [lundi] => 1234 [mardi] => 2222 [mercredi] => 4444 [jeudi] => 6666 )
```

PHP - Initiation

Les tableaux

tableau*[*cle*][*indice*] = *valeur

Associe ***valeur*** à l'indice ***indice*** du tableau contenu à la clé ***cle***.
(On peut combiner autant de tableaux qu'on le souhaite, de tout type.)

```
$tab1 = [1,2,3,4];
$tab2 = [5,6,7,8];
$tableau = ['lundi' => $tab1, 'mardi' => $tab2];
$tableau['mardi'][2] = 9;
```

array_push(*tableau*, *valeur(s)*)

Cette fonction est équivalent à ***tableau*[] = *valeur***.
La principale différence est qu'il est possible d'ajouter plusieurs entrées en une seule instruction en les séparant par des **,** (virgules).

array_unshift(*tableau*, *valeur(s)*)

Cette fonction permet également d'insérer des valeurs mais en début de tableau. Cela aura pour effet de recalculer et décaler tous les indices !

PHP - Initiation

Les tableaux

IV.6.2) Supprimer des données

array_pop(*tableau*)
Renvoi et supprime le dernier élément du tableau.

array_shift(*tableau*)
Renvoi et supprime le premier élément du tableau. Recalcul et décale les indices afin qu'ils commencent à zéro.

```
$fruits = ['orange', 'banane', 'clémentine', 'pomme'];

$sup_deb = array_shift($fruits);
echo $sup_deb;          // va afficher : orange
print_r($fruits);       // va afficher : Array ( [0] => banane [1] => clémentine [2] => pomme )

$sup_fin = array_pop($fruits);
echo $sup_fin;          // va afficher : pomme
print_r($fruits);       // va afficher : Array ( [0] => banane [1] => clémentine )
```

PHP - Initiation

Les tableaux

IV.6.3) Fonctions utiles

count(*tableau*)
Renvoi tout simplement le nombre d'éléments d'un tableau.

in_array(*valeur*, *tableau*)
Renvoi **true** si la valeur ***valeur*** est trouvée dans ***tableau***, **false** dans le cas contraire.

array_search(*valeur*, *tableau*)
Renvoi la **clé** si la valeur ***valeur*** est trouvée dans ***tableau***, **false** dans le cas contraire.

```
$fruits = ['orange', 'banane', 'clémentine', 'pomme'];
$reponse = in_array('banane', $fruits);
var_dump($reponse);              // va afficher : bool(true)
$reponse = array_search('clémentine', $fruits);
var_dump($reponse);              // va afficher : int(2)
```

PHP - Initiation

Les tableaux

explode(*delimiteur*, *chaîne*)
Retourne un tableau de valeurs crées en découpant ***chaîne*** suivant le paramètre ***delimiteur***.

implode(*delimiteur*, *tableau*)
Retourne un chaîne de caractères constituée de toutes les valeurs de ***tableau*** séparées par le paramètre ***delimiteur***.
(Ce dernier est optionnel)

```
$fruits = ['orange', 'banane', 'clémentine', 'pomme'];

$chaine = implode(',',$fruits);
print_r($chaine);
// va afficher : orange,banane,clémentine,pomme

$tableau = explode(',', $chaine);
print_r($tableau);
// va afficher : Array ( [0] => orange [1] => banane [2] => clémentine [3] => pomme )
```

PHP - Initiation

Les tableaux

sort(*tableau*, *option*)
Va modifier l'ordre des éléments du ***tableau*** selon le paramètre ***option***. Si ce dernier est omi, le tri se fait sur les valeurs, de la plus petite à la plus grande. (option **SORT_REGULAR**)

```
$fruits = ['orange', 'banane', 'clémentine', 'pomme'];
sort($fruits);
print_r($fruits);
// va afficher : Array ( [0] => banane [1] => clémentine [2] => orange [3] => pomme )
```

Quelques options de tri possible :

- **SORT_NUMERIC** compare les éléments numériquement
- **SORT_STRING** compare les éléments comme des chaînes
- **SORT_NATURAL** compare les éléments "naturellement"

```
$nombres = ['img1.jpg', 'img3.jpg', 'img12.jpg', 'img2.jpg'];
sort($nombres, SORT_NATURAL);
print_r($nombres);
// va afficher : Array ( [0] => img1.jpg [1] => img2.jpg [2] => img3.jpg [3] => img12.jpg )
```