COURS 04 : LES TABLEAUX

LES CHAINES DE CARACTÈRES

Les chaînes de caractères sont en fait des tableaux de caractère. Testez :

```
<?PHP
  $mot ="informatique";
  echo ( $mot[0] ."\n");
  echo ( $mot[1] ."\n");
 ?>
```

$mot[0] → signifie le premier caractère du mot, on parle du caractère qui se trouve à l'**indice 0**

$mot[1] → signifie le deuxième caractère du mot, on parle du caractère qui se trouve à l'**indice 1**

Si on veut afficher le dernier caractère du mot, on doit calculer la longueur du mot grâce à la fonction **strlen($mot) ,** puis prendre l'indice se trouvant juste avant c'est à dire : **strlen($mot) – 1**

$mot[strlen($mot)] → provoquera une erreur PHP Notice: Uninitialized string offset: 12, en effet les indices commençant à 0, l'indice 12 n'existe pas.

$mot[strlen($mot) – 1] → affichera bien le dernier caractère

Comment affichez ce mot caractère par caractère en sautant une ligne entre chaque caractère ?

Boucle **while :**

boucle **for :**

Même question mais en partant du dernier caractère jusqu'au premier ?

LES TABLEAUX

Jusqu'à présent, hormis les chaînes de caractères, nous avons manipulés des variables "simples" : au nom d'une variable correspond une et une seule valeur à un instant *t* donné.

Or il arrive souvent d'avoir à gérer un **ensemble** de valeurs de façon uniforme. Nous pouvons alors les stocker dans une **structure conteneur**, appelée **collection**.

Exemple, un professeur souhaite trouver un moyen de représenter les moyennes semestrielles de ses étudiants. Il y a 35 étudiants, va-t-il déclarer 35 variables ? non , il va utiliser une collection.

CARACTÉRISTIQUE DES TABLEAUX EN PHP

- Les tableaux ont une taille infinie.
- Il est possible de déclarer un tableau mais ce n'est pas obligatoire.

Avec déclaration du tableau	Sans déclaration du tableau
$lesSaisons = array( ) ; $lesSaisons[0] = "hiver" ; $lesSaisons[1] = "printemps" ;	$lesSaisons[0] = "hiver" ; $lesSaisons[1] = "printemps" ;

- Il est possible de déclarer un tableau tout en l'initialisant.
  Exemple : $lesSaisons = array("hiver"," printemps ","été","automne");

- Il est possible de connaître le nombre d'éléments contenus dans un tableau à l'aide de la fonction sizeof ou count
  Exemple :

```
print ( sizeof( $lesSaisons )) ; // affichera 4
print ( count( $lesSaisons )) ; // affichera 4
```

Dans notre exemple, la liste est indicée par des entiers, **la valeur du premier indice est zéro**. Mais rien n'interdit de démarrer les indices à -1 ou +1, -40, +200... mieux, les indices peuvent être des chaînes de caractères !

indice	**0**	**1**	**2**	**3**
valeur	hiver	printemps	été	automne

- Pour afficher le premier élément du tableau
  print($lesSaisons[0]); //affichera hiver

- Pour visualiser le contenu du tableau
  print_r($lesSaisons);//affichera Array

```
(
    [0] => hiver
    [1] => printemps
    [2] => été
    [3] => automne
)
```

- Pour manipuler les éléments d'un tableau, il faut parcourir le tableau et donc utiliser des instructions de boucle.
  Exemple : Afficher un tableau, c'est afficher chaque élément du tableau.

```
for ($i = 0 ;  $i  <  count( $lesSaisons) ;  $i ++ ){
        echo ($lesSaisons[$i] );
}
```

MANIPULATION D'UN TABLEAU

CRUD est un acronyme qui désigne les 4 opérations fondamentales sur des données. Nous les présentons ici, avec une traduction PHP sur une donnée de type tableau (array) :

- **Create** (création)
- On retient 2 contextes : création d'un tableau, et **ajout** d'un élément dans un tableau.
  - Création d'un tableau : On utilise la fonction `array()`, nous pouvons également initialiser le tableau :

```
$notes=array(); // création d'une liste vide
$notes=array(16,2);// création d'une liste à 2 éléments
```

  - Ajout d'un élément **en fin de liste** :

- Par un indice :

```
$notes[count($notes)]=2;
```

  On ajoute en fin de liste un élément ayant comme valeur 2.

- Sans indice :

```
$notes[]=12;
```

  On ajoute en fin de liste un élément ayant comme valeur 2

- **Retreive** (recherche)

  Consiste ici à évaluer une valeur du tableau. Pour cela nous avons besoin de deux informations : **nom de la variables collection** (ici `$notes`) et d'une **clé** (ici un indice de type nombre entier).

```
echo  $notes[1]; //affichera 2
```

  Que se passe-t-il s'il n'y a pas d'élément pour un indice donné ?

```
$x = $notes[9999];
```

  Pour bon nombre de langage, il s'agirait d'une erreur du développeur. PHP est plus souple, et renvoie **`FALSE`**, qui sera affectée à `$x` dans notre exemple. Il affichera en fonction de la configuration de l'interpréteur PHP un message d'erreur :
  Undefined offset: 10 → ce qui signifie que la variable se situant à l'indice 10 n'est pas définie

- **Update** (mise à jour)

  Typiquement une démarche en 3 temps :

```
// Exemple nous souhaitons ajouter 12 points à la deuxième note
// de notre liste (de valeur 2 à la valeur 14).

// 1: Extraire la valeur à modifier
$noteAChanger = $notes[1];
// 2: Modifier la valeur
$noteAChanger = $noteAChanger + 12;
// 3: Réinsérer la nouvelle valeur
$notes[1] = $noteAChanger;
```

  Nous pouvons réaliser cela en une seule instruction :

```
$notes[1] = $notes[1] + 12;
```

- **Delete** (suppression)

Pour supprimer un tableau entier, on utilise la fonction `unset` :

```
unset($notes);
print($notes); // $notes ne vaut plus rien
```

Pour supprimer un élément du tableau, on appel `unset` avec le bon indice :

```
$acteurs = array('Dupont','Dupond','Tintin');
unset($acteurs[1]);
/* Cela va produire un tableau comme celui-ci
 $acteurs = array(0 => 'Dupont', 2 => 'Tintin');
  et non pas array(0 => 'Dupont', 1 =>'Tintin');
*/
print($acteurs[0]."\n");  // affiche Dupont
print($acteurs[1]."\n");  // n'affiche rien (FALSE)
print($acteurs[2]."\n");  // affiche Tintin
```

Fonctions pratiques

Il y a toute une panoplie de fonctions pratiques pour travailler avec les tableau : array-functions.
Les principales sont les suivantes :

- **sort** :`void sort (array array)`

sort() trie le tableau *`array`*. Les éléments seront triés du plus petit au plus grand.

```
<?php
$fruits = array("papaye","orange","banane","ananas");
sort($fruits);
print_r($fruits); //va afficher 0==>ananas , 1==>banane, 2==>orange, 3==>papaye
?>
```

- **rsort:** `void rsort (array array)`

rsort() effectue un tri en ordre décroissant (du plus grand au plus petit).

- **array_sum** (array array)

retourne la somme des valeurs du tableau array , sous forme d'un entier ou d'un nombre à virgule flottante.

- **max** (array numbers)

retourne la plus grande valeur numérique parmi les valeurs passées en paramètre.

```
<?PHP
 $lesPrix=array(2,12,5,100,2,0,3,-5);
 $maxPrix=max($lesPrix);
 print($maxPrix); // va afficher 100
?>
```

- **min** (array numbers)

retourne la plus petite valeur numérique parmi les valeurs passées en paramètre.

- **print_r**(array array) affiche les valeurs d'un tableau. Les valeurs seront affichées dans un format permettant de voir les indices et les éléments