

Cours Web n°6

Introduction au langage PHP (suite)

Pierre Senellart (pierre@senellart.com)

Pierre Yger (yger@unic.cnrs-gif.fr)



Semaine du 22 octobre 2007

Plan du cours

- 1 Les Tableaux
- 2 Les fonctions
- 3 PHP et HTTP
- 4 Références
- 5 Applications

- Dans un langage de programmation, un **tableau** est une suite de valeurs référencées par une unique variable.
- La taille d'un tableau est déterminée par le nombre de valeurs qu'il contient.
- Les tableaux peuvent être :
 - indicés** et les valeurs sont référencées par leur position en partant de 0
 - associatifs** et les valeurs sont référencées par des noms ou clefs

- Un tableau peut contenir des valeurs de types différents : entier, chaîne de caractères, etc.

Exemple

```
$tab[0] = "1er element";  
$tab[1] = "2e element";  
$tab[2] = 120;
```

- En PHP, l'affectation d'un indice à un nouvel élément est automatique : il correspond à la première cellule vide.

Exemple

```
$tab2[] = "1er element";  
$tab2[] = "2e element";  
$tab2[] = 120;
```

- L'initialisation d'un tableau peut également se faire à l'aide de la fonction `array`

Exemple

```
$tab3 = array("1er element", "2e element", 120);
```

- Les indices numériques sont remplacés par des chaînes de caractères appelées **clefs**.
- Pour un tableau donné, toutes les clefs doivent être différentes.
- La fonction `array` peut également être utilisée pour initialiser le tableau.

Exemple

```
$prenom["Belmondo"] = "Jean-Paul";  
$prenom["Delon"] = "Alain";  
$prenom["Deneuve"] = "Catherine";  
$prenom = array("Balasko" => "Josiane", "Bourvil" => "");
```

- Le parcours d'un tableau associatif est plus complexe que celui des tableaux indicés : on ne peut pas se baser sur l'ordre des indices pour effectuer une boucle simple.
- On peut utiliser un curseur sur le tableau : sorte de flèche indiquant l'élément du tableau actuellement visé.
- Les fonctions `next` et `prev` permettent de déplacer le curseur initialement positionné sur le premier élément du tableau.
- Les fonctions `key` et `current` renvoient respectivement la clef et la valeur de l'élément courant.

Exemple (Afficher toutes les personnes de la liste prenom)

```
echo "Nom=".key($prenom).  
    " Prenom=".current($prenom)."\n";  
  
while(next($prenom)) {  
    echo "Nom=".key($prenom). " Prenom=".current($prenom);  
}
```

- Il est cependant plus facile de parcourir un tableau associatif à l'aide de l'instruction `foreach`.

Exemple

```
foreach ($prenom as $cle => $valeur) {  
    echo "Nom=$cle Prenom=$valeur\n";  
}
```

- Cette instruction peut également être utilisée pour des tableaux indicés.

Exemple

```
foreach ($tableau as $valeur) {  
    ...  
}
```

Plan du cours

- 1 Les Tableaux
- 2 Les fonctions**
- 3 PHP et HTTP
- 4 Références
- 5 Applications

- Pour faciliter la visibilité d'un script PHP et pour rendre automatique l'exécution de certaines tâches répétitives, on peut définir des **fonctions**.
- Comme en mathématiques, les fonctions prennent en argument (ou non) des variables et retourne une valeur à l'aide de l'instruction `return`
- La construction `list (...)` peut être utilisée pour récupérer dans plusieurs variables différentes une valeur de retour qui est un tableau.

Exemple

```
function Kenshin() {  
    return array ("Kenshin Le Vagabond", "Nobuhiro Watsuki", 28);  
}  
  
function Addition($x,$y) {  
    $somme = $x+$y;  
    return $somme;  
}  
  
$z=Addition($x,$y);  
list ($a,$b,$c)=Kenshin();
```

`abs(n)` renvoie la valeur absolue de *n* qui peut être un entier ou un réel.

`ceil(f)` renvoie le plus petit entier supérieur ou égal à *f*.

`empty(v)` renvoie faux si la variable *v* est définie et a une valeur non nulle, vrai sinon.

`isset(v)` indique si la variable *v* est définie.

`floor(f)` renvoie le plus grand entier inférieur ou égal à *f*.

`max(v1,...,vn)` renvoie la plus grande des valeurs passées en paramètre.

`min(v1,...,vn)` renvoie la plus petite des valeurs passées en paramètre.

`rand()` renvoie une valeur aléatoire.

`round(f)` renvoie l'entier le plus proche de *f*.

`explode(sep,chaine)` divise *chaine* en valeurs séparées par *sep* et renvoie le tableau de ces valeurs.

`implode(sep,tableau)` renvoie une chaîne de caractères avec les valeurs de *tableau* séparées par *sep*.

Exemple

```
$fruits = "abricot|kiwi|pomme|fraise|banane";  
$listeFruits = explode('|',$fruits);  
$fruits2 = implode(' ', $listeFruits);  
echo $fruits2;
```

⇒ abricot kiwi pomme fraise banane

`strstr(ch1,ch2)` renvoie le contenu de *ch1* à partir de la première occurrence de *ch2*. Si *ch2* n'apparaît pas dans *ch1*, elle renvoie faux.

`strlen(ch)` renvoie la longueur de la chaîne *ch*.

`substr(ch,deb,long)` renvoie la sous-chaîne de *ch* de longueur *long* à partir de *deb*.

Exemple

```
$fruits = "abricot|kiwi|pomme|fraise|banane";  
$tmp = strchr($fruits,'|');  
echo $tmp;
```

⇒ |kiwi|pomme|fraise|banane

```
$tmp2 = substr($fruits,8,4);  
echo $tmp2;
```

⇒ kiwi

`date("d/m/Y")` renvoie la date courante formatée 22/10/2007

- Il existe d'autres options de formatage, par exemple :

- `h` heure sur 12 heures

- `H` heure sur 24 heures

- `y` année sur deux chiffres

- etc.

- `is_array(tab)` permet de savoir si une variable donnée est un tableau.
- `count(tab)` renvoie le nombre d'éléments du tableau.
 - `sort(tab)` trie le tableau sur les valeurs en ordre ascendant.
 - `rsort(tab)` trie le tableau sur les valeurs en ordre descendant.
 - `ksort(tab)` trie le tableau associatif sur la clef en ordre ascendant.
 - `krsort(tab)` trie le tableau associatif sur la clef en ordre descendant.
- `max(tab)` renvoie la plus grande valeur du tableau.
- `min(tab)` renvoie la plus petite valeur du tableau.

Exemple

```
$listeFruits = array('abricot', 'pomme', 'kiwi', 'fraise', 'banane');  
echo sort($listeFruits);
```

⇒ abricot banane fraise kiwi pomme

```
echo rsort($listeFruits);
```

⇒ pomme kiwi fraise banane abricot

```
$prenom=array('Delon'=>'Alain', 'Deneuve'=>'Catherine',  
             'Belmondo'=>'Jean-Paul');
```

```
foreach (ksort($prenom) as $cle => $valeur) {  
    echo "$valeur ";  
}
```

⇒ Jean-Paul Alain Catherine

Plan du cours

- 1 Les Tableaux
- 2 Les fonctions
- 3 PHP et HTTP**
- 4 Références
- 5 Applications

- Les paramètres HTTP peuvent être récupérées en PHP grâce au **tableau associatif** `$_REQUEST` .
- Les valeurs de ce tableau peuvent être des variables simples ou des tableaux indicés : ces derniers sont les paramètres à choix multiples dont on a suffixé le nom de `[]` dans le code XHTML.

Exemple

```
echo "<p>Votre login est : " . $_REQUEST["login"] . "</p>";
echo "<p>Vous avez coché les genres : ";
for($i=1;$i<=count($_REQUEST['genre']);$i=$i+1) {
    echo $_REQUEST['genre'][$i] . " ";
}
echo "</p>";
```

Plan du cours

- 1 Les Tableaux
- 2 Les fonctions
- 3 PHP et HTTP
- 4 Références**
- 5 Applications

- <http://www.php.net/>
- *Pratique de MySQL et PHP*, Philippe Rigaux, O'Reilly

Plan du cours

- 1 Les Tableaux
- 2 Les fonctions
- 3 PHP et HTTP
- 4 Références
- 5 Applications**

- Reproduire le comportement du script de test utilisé lors du cours n°4 : il construit un tableau de l'ensemble des variables passées en paramètres lors de la requête HTTP. Attention aux variables tableau !
- Écrire une fonction prenant trois arguments \$a, \$b et \$c et renvoyant la chaîne de caractères obtenue en remplaçant dans \$a toutes les occurrences de \$b par \$c. Tester cette fonction avec des valeurs de paramètres HTTP.
- Essayer de construire une fonction reproduisant le comportement de la fonction prédéfinie sort pour trier un tableau indicé. Quelles stratégies peut-on adopter ?