PHP : Les tableaux

L2 MPCIE - UE Développement Web

David Lesaint

david.lesaint@univ-angers.fr

Janvier 2019

Les tableaux

Fonctions natives :

- Création de tableaux.
- Lecture des éléments.
- Ajout et retrait d'éléments.
- Tri des éléments.
- Sélection d'éléments.
- Fonctions sur tableau.

Création de tableaux avec fonction `array()`

`$tab=array(v0,...,vN);`

Créée un tableau indicé à `N+1` éléments :

- Le 1er élément de valeur `v0` a pour indice 0.
- Syntaxe alternative : `$tab=[v0,...,vN];`
- Accès au `i`+1-ième élément : `$tab[i]`.

`$tab=array("k1"=>v1,...,"kN"=>vN);`

Créée un tableau associatif à `N` éléments :

- Chaque élément est une paire clé `"k"` - valeur `v`.
- Pas de notion d'ordre entre les éléments.
- Accès par clé aux éléments : `$tab["k"]`.
- Syntaxe alternative :
  `$tab=["k1"=>v1,...,"kN"=>vN];`

Tableaux multidimensionnels

Sous la forme de tableaux de tableaux

`$tab=array(array(...),...,array(...));`

- Alternative : `$tab=[[...],...,[...]];`

`$tab=array("k1"=>array(...),...,"kN"=>array(...));`

- Alternative : `$tab=["k1"=>[...],...,"kN"=>[...]];`

Accès : `$tab[a][b]` où `a` et `b` sont des indices ou clés selon le cas.

exemple5-1.php

```
1  $tab=[  ["A-0","A-1","A-2"],
2          ["B-0","B-1","B-2"],
3          ["C-0","C-1","C-2"],
4          ["D-0","D-1", "D-2"]];
5  echo "<h3>Tableau multidimensionnel</h3><table border='1' width=\"100%\">
<tbody>";
6  for ($i=0;$i<count($tab);$i++) {
7       echo "<tr>";
8       for($j=0;$j<count($tab[$i]);$j++)
9            echo "<td><h3> .. ",$tab[$i][$j]," .. </h3></td>";
10      echo "</tr>";
11 }
12 echo " </tbody> </table> ";
13 print_r($tab);
```

Suites de nombres/lettres

`array range(int m, int M)`

Créée un tableau indicé contenant tous les entiers de `m` à `M`.

`array range(char c, char C)`

Créée un tableau indicé contenant tous les caractères de `c` à `C` selon le code ASCII.

exemple5-2.php

```php
// Suite de nombres de 1 à 10
$tabnombre= range(1,10);
print_r($tabnombre);
echo "<hr>";
// Suite de lettres de a à z avec une boucle
for($i=97;$i<=122;$i++) {
      $tabalpha[$i-96]=chr($i);
}
print_r($tabalpha);
echo "<hr>";
//Suite des caractères de Y à b avec range()
$tabalpha2 = range("Y","b");
print_r($tabalpha2);
```

Transformations de chaînes en tableaux

`array explode(string sep, string $ch [, int N])`

Décompose la chaîne `$ch` en tableau de mots (sous-chaînes) selon le séparateur `sep` fourni. `N` est le nombre maximum de mots recherchés.

Décompte des éléments

`int count(array $tab)` ou `int sizeof(array $tab)`

Retourne le nombre d'éléments.

Pour tableau "muti-dimensionnel"

```php
// Comptage du nombre d'éléments
$tab=array("Bonjour","Web",array("1-0","1-1","1-2"),
     1970,2013,array("3-0","3-1","3-2","3-3"));
echo "Le tableau \$tab a ",count($tab)," éléments <br />";
//ou encore: echo "Le tableau \$tab a ",sizeof($tab)," éléments <br />";
// Comptage du nombre de valeurs
$nb_val=0;
for ($i=0;$i<count($tab);$i++) {
 if(gettype($tab[$i])=="array") {
  $nb_val+=count($tab[$i]);
 } else {
  $nb_val++;
 }
 }
echo "Le tableau \$tab a ",$nb_val," valeurs <br />";
```

Décompte des valeurs

`array array_count_values(array $tab)`

Retourne un tableau associatif ayant pour clés les valeurs de `$tab` et pour valeur le nombre d'occurrences de chaque valeur dans `$tab`.

- Ne s'applique qu'aux tableaux de nombres et/ou chaînes.

exemple5-4.php

```php
$tab= ["PHP","JavaScript","PHP","ASP","PHP","ASP"];
$result=array_count_values($tab);
echo "Le tableau \$tab contient ",count($tab)," éléments <br>";
echo "Le tableau \$tab contient ",count($result)," valeurs différentes <br>";
print_r($result);
```

Lecture des éléments avec boucles `for` et `while`

Avec boucle `for`

```php
$montab=array("Paris","London","Brüssel");
for ($i=0;$i<count($montab);$i++) {
  echo "L'élément $i est $montab[$i]<br />";
}
```

Avec boucle `while`

```php
$montab=array("Paris","London","Brüssel");
$i=0;
while(isset($montab[$i])) {
  echo "L'élément $i est $montab[$i]<br />";
  $i++;
}
```

Lecture des éléments avec `while()` et `each()`

L'expression `$e=each($tab)`

- Est utilisée comme condition de boucle `while`.
- Est évaluée à `FALSE` si le pointeur interne est positionné en fin du tableau `$tab`.
- Sinon fait avancer le pointeur interne sur l'élément suivant.
- `$e` est un tableau d'informations sur l'élément courant :
  - `$e[0]` ou `$e["key"]` contient l'indice/clé de l'élément.
  - `$e[1]` ou `$e["value"]` contient la valeur de l'élément.

`reset($tab)` positionne le pointeur interne en début de tableau.

Exemple avec `each`

exemple5-9.php

```php
1  //******Lecture d'un tableau indicé"******
2  $montab=array("Paris","London","Brüssel");
3  //Ajout d'un élément au tableau
4  $montab[9]="Berlin";
5  //Lecture des éléments
6  reset($montab);
7  while($element=each($montab)) {
8   echo "L'élément d'indice $element[0] a la valeur $element[1]<br />";
9  }
10 echo "<hr>";
11 //******Lecture d'un tableau associatif******
12 $montab=array("France"=>"Paris","Great
Britain"=>"London","België"=>"Brüssel");
13  //Ajout d'un élément au tableau
14 $montab["Deutschland"]="Berlin";
15 //Lecture des éléments
16 reset($montab);
17 while($element=each($montab)) {
18  echo "L'élément de clé {$element['key']} a la valeur {$element['value']}<br
/>";
19 }
```

Lecture des éléments avec `list()` et `each()`

L'expression `list($x1,...,$xN)=$tab`

- Ne s'applique qu'aux tableaux indicés `$tab`.
- Affecte la `i`-ième valeur de `$tab` à `$xi`.
- Ne modifie pas le pointeur interne de `$tab`.
- Utilisable comme condition de boucle `while` avec la syntaxe `list($idx,$val)=each($tab)`.

Filtrage des valeurs

`list($x1,,,$x4)=$tab` récupère dans `$x1` et `$x4` les éléments d'indice 0 et 3.

Exemple avec `list`

exemple5-11.php

```php
$tab=array("Paris","London","Brüssel");
list($x,$y) = $tab;
echo "Les deux premiers éléments sont : $x et $y <hr />";
$tab=array("Paris","London","Brüssel");
while(list($indice,$valeur) = each($tab)) {
 echo "L'élément d'indice <b>$indice</b> a la valeur <b>$valeur</b><br>";
}
echo"<hr />";
```

Lecture des éléments avec `foreach()`

`foreach($tab as $val){bloc}`

- Pour tableaux indicés.
- La variable `$val` contiendra successivement chacune des valeurs du tableau `$tab`.

`foreach($tab as $key=>$val){bloc}`

- Pour tableaux associatifs.
- La paire de variables (`$key`,`$val`) correspondra successivement à chaque élément (clé,valeur) du tableau `$tab`.

Exemple avec `foreach`

exemple5-12.php

```php
//*****************************************************
//Lecture de tableau indicé sans récupération des indices
//*****************************************************
$tab=array("Paris","London","Brüssel");
echo "<H3>Lecture des valeurs des éléments </H3>";
foreach ($tab as $ville) {
 echo "<b>$ville</b> <br>";
}
echo"<hr>";
//*****************************************************
//Lecture de tableau indicé avec récupération des indices
//*****************************************************
echo "<h3>lecture des indices et des valeurs des éléments </h3>";
foreach ($tab as $indice=>$ville) {
 echo "L'élément d'indice <b>$indice</b> a la valeur <b>$ville</b><br>";
}
echo"<hr>";
//*****************************************************
//Lecture de tableau associatif avec récupération des clés
//*****************************************************
$tab2=array("France"=>"Paris","Great Britain"=>"London","België"=>"Brüssel");
echo "<h3>lecture des clés et des valeurs des éléments</h3>";
foreach ($tab2 as $cle=>$ville) {
 echo "L'élément de clé <b>$cle</b> a la valeur <b>$ville</b> <br>";
}
echo"<hr>";
```

Extraction d'éléments avec `array_slice()`

`array array_slice(array $tab, int i, int n)`

Retourne un sous-tableau de `$tab` différent selon le signe et la valeur de `i` et `n` :

1. `array_slice($tab,2,3)` : les 3 premiers éléments à partir de celui d'indice 2.
2. `array_slice($tab,2,-3)` : tous les éléments à partir de celui d'indice 2 sauf les 3 derniers.
3. `array_slice($tab,-2,3)` : les 3 éléments précédant l'avant-dernier.
4. `array_slice($tab,-3,-2)` : les éléments d'indices virtuels négatifs entre `-3` compris et `-2` non compris.

exemple5-14.php

Ajout et retrait d'éléments

`int array_push($tab,v1,...,vN)`

- Ajoute `v1,...,vN` en fin de `$tab` et en retourne la taille.
- Indexation de `v1,...,vN` à partir de `count($tab)`.

`mixed array_pop($tab)`

- Supprime et retourne le dernier élément de `$tab` s'il existe, `NULL` sinon.

`void unset(mixed $tab [, mixed $...])`

Utilisé avec un élément de tableau comme argument :

- Supprime l'élément.
- Sans conséquence sur les indices/clés des éléments restants.

Ajout et retrait d'éléments

`int array_unshift($tab,v1,...,vN)`

- Ajoute `v1,...,vN` en début de `$tab` et en retourne la taille.
- Ré-indexation à partir de 0 avec décalage-droite pour éléments indicés, clés inchangées pour les autres.

`mixed array_shift($tab)`

- Supprime et retourne le premier élément de `$tab` s'il existe, `NULL` sinon.
- Ré-indexation à partir de 0 avec décalage-gauche pour éléments indicés, clés inchangées pour les autres.

`array array_unique($tab)`

Supprime les valeurs en doublons et ne conserve que la "dernière" occurrence.

- Sans conséquence sur les indices/clés des éléments restants.

Exemple

exemple5-15.php

```php
$tab= array(800,1492, 1515, 1789); print_r($tab);
echo "<hr />";
// Ajout au début du tableau
$poitiers=732;
$nb=array_unshift($tab,500,$poitiers);
echo "Le tableau \$tab a maintenant $nb éléments <br>"; print_r($tab);
echo "<hr />";
// Ajout à la fin du tableau
$armi=1918;
$newnb=array_push($tab,1870,1914,$armi);
echo "Le tableau \$tab a maintenant $newnb éléments <br>"; print_r($tab);
echo "<hr />";
// Suppression du dernier élément
$suppr= array_pop($tab);
echo "Le tableau \$tab a perdu l'élément $suppr <br>"; print_r($tab);
echo "<hr />";
// Suppression du premier élément
$suppr= array_shift($tab);
echo "Le tableau \$tab a perdu l'élément $suppr <br>"; print_r($tab);
echo "<hr />";
// Suppression de l'élément d'indice 4
unset($tab[4]);
echo "L'élément d'indice 4 a été supprimé <br>"; print_r($tab);
```

Tri d'éléments

Différentes méthodes

- Tri sur tableaux indicés ou sur tableaux associatifs.
- Tri sur les valeurs ou sur les clés.
- Critères de tri prédéfinis (ordre alphabétique, ordre ASCII, ...) ou personalisés.
- Préservation ou non des indices/clés.

Tri sur tableaux indicés

Selon l'ordre ASCII

- `bool sort(array &$tab)` : tri des valeurs en ordre croissant. Perte des correspondances indices-valeurs.
- `bool rsort(array &$tab)` : tri des valeurs en ordre décroissant. Perte des correspondances indices-valeurs.
- `array array_reverse(array $tab)` : inverse l'ordre des valeurs. Perte des correspondances indices-valeurs.

exemple5-18.php

Tableau indicé d'origine
Array ([0] => Blanc2 [1] => Jaune [2] => rouge [3] => Vert [4] => Bleu [5] => Noir [20] => Blanc10)

Tri en ordre ASCII sans sauvegarde des indices
Array ([0] => Blanc10 [1] => Blanc2 [2] => Bleu [3] => Jaune [4] => Noir [5] => Vert [6] => rouge)

Tri en ordre ASCII inverse sans sauvegarde des indices
Array ([0] => rouge [1] => Vert [2] => Noir [3] => Jaune [4] => Bleu [5] => Blanc2 [6] => Blanc10)

Inversion de l'ordre des éléments
Array ([0] => Blanc10 [1] => Noir [2] => Bleu [3] => Vert [4] => rouge [5] => Jaune [6] => Blanc2)

Tri sur tableaux indicés ou associatifs

Selon l'ordre naturel

- `bool natsort(array &$tab)` : tri des valeurs selon l'ordre "naturel" (chiffres avant majuscules avant minuscules). Préservation des correspondances indices/clés-valeurs.
- `bool natecasesort(array &$tab)` : comme `natsort` mais insensible à la case.

Boucle `foreach` indispensable pour itérer selon le nouvel ordre.

exemple5-19.php

Tableau indicé d'origine
Array ([0] => Blanc2 [1] => Jaune [2] => rouge [3] => Vert [4] => Bleu [5] => Blanc10 [6] => 1ZZ [7] => 2AA)

Tri en ordre naturel avec sauvegarde des indices
Array ([6] => 1ZZ [7] => 2AA [0] => Blanc2 [5] => Blanc10 [4] => Bleu [1] => Jaune [3] => Vert [2] => rouge)

Tri en ordre naturel insensible à la casse avec sauvegarde des indices
Array ([6] => 1ZZ [7] => 2AA [0] => Blanc2 [5] => Blanc10 [4] => Bleu [1] => Jaune [2] => rouge [3] => Vert)

Tri personnalisé sur tableaux indicés

Sur la base d'une fonction binaire numérique à définir (le critère de tri)

`int moncritere(mixed $x, mixed $y)` doit renvoyer :

- -1 (ou nombre négatif) si `$x` est "inférieur" à `$y`.
- 0 si `$x` est "égal/équivalent" à `$y`.
- 1 (ou nombre positif) si `$x` est "supérieur" à `$y`.

Fonction utilisée comme argument de `bool usort(array &$tab, callable $f)`

```
usort($tab,"moncritere");
```

- Trie `$tab` selon la fonction `moncritere()`.
- Perte des correspondances indices-valeurs.

Exemple

exemple5-20.php

```php
// Définition de la fonction de tri
function long($mot1,$mot2) {
if(strlen($mot1)==strlen($mot2))
        return 0;
elseif(strlen($mot1)>strlen($mot2))
        return -1;
else
        return 1;
}
// Fonction de tri équivalente
function long1($mot1,$mot2) { return strlen($mot2)<=>strlen($mot1); }
// Tableau à trier
$tab=["Blanc","Jaune","rouge","Vert","Orange","Noir","Emeraude"];
// Utilisation de la fonction de tri
echo "Tableau initial<br />";
print_r($tab);
usort($tab,"long");
echo "<br />Tableau trié selon la longueur décroissante des mots<br />";
print_r($tab);
```

exemple5-20.php

Tableau initial
Array ([0] => Blanc [1] => Jaune [2] => rouge [3] => Vert [4] => Orange [5] => Noir [6] => Emeraude)
Tableau trié selon la longueur décroissante des mots
Array ([0] => Emeraude [1] => Orange [2] => Blanc [3] => Jaune [4] => rouge [5] => Vert [6] => Noir)

Tri des valeurs sur tableaux associatifs

Avec préservation des correspondances clés-valeurs

- `bool asort(array &$tab)` : tri ASCII des valeurs en ordre croissant.
- `bool arsort(array &$tab)` : tri ASCII des valeurs en ordre décroissant.
- `bool uasort(array $tab, callable $f)` : tri personnalisé des valeurs avec `$f`.

exemple5-22.php

Tableau associatif d'origine
Array ([w2] => Blanc2 [y] => Jaune [r] => rouge [g] => Vert [blu] => Bleu [bla] => Noir [w10] => Blanc10)

Tri en ordre ASCII des valeurs avec sauvegarde des clés
Array ([w10] => Blanc10 [w2] => Blanc2 [blu] => Bleu [y] => Jaune [bla] => Noir [g] => Vert [r] => rouge)

Tri en ordre ASCII inverse avec sauvegarde des clés
Array ([r] => rouge [g] => Vert [bla] => Noir [y] => Jaune [blu] => Bleu [w2] => Blanc2 [w10] => Blanc10)

Tri en ordre naturel avec sauvegarde des clés
Array ([w2] => Blanc2 [w10] => Blanc10 [blu] => Bleu [y] => Jaune [bla] => Noir [g] => Vert [r] => rouge)

Tri en ordre naturel insensible à la casse avec sauvegarde des clés
Array ([w2] => Blanc2 [w10] => Blanc10 [blu] => Bleu [y] => Jaune [bla] => Noir [r] => rouge [g] => Vert)

Tri des clés sur tableaux associatifs

Avec préservation des correspondances clés-valeurs

- `bool ksort(array &$tab)` : tri ASCII des clés en ordre croissant.
- `bool krsort(array &$tab)` : tri ASCII des clés en ordre décroissant.
- `bool uksort(array $tab, callable $f)` : tri personnalisé des clés avec `$f`.

exemple5-23.php

Tableau associatif d'origine
Array ([w2] => Blanc2 [y] => Jaune [r] => rouge [g] => Vert [blu] => Bleu [bla] => Noir [w10] => Blanc10)

Tri en ordre ASCII des clés
Array ([bla] => Noir [blu] => Bleu [g] => Vert [r] => rouge [w10] => Blanc10 [w2] => Blanc2 [y] => Jaune)

Tri en ordre ASCII inverse des clés
Array ([y] => Jaune [w2] => Blanc2 [w10] => Blanc10 [r] => rouge [g] => Vert [blu] => Bleu [bla] => Noir)

Tri selon la longueur des clés
Array ([w10] => Blanc10 [blu] => Bleu [bla] => Noir [w2] => Blanc2 [y] => Jaune [r] => rouge [g] => Vert)

Sélection d'éléments

`array array_filter($tab, callable $f)`

Retourne le tableau d'éléments de `$tab` acceptés par la *call-back* `$f`

- bool `$f(mixed $var)` est appelée avec chaque élément `$var` de `$tab` et doit retourner `TRUE` ssi elle accepte l'élément.
- Préserve les correspondances clés-valeurs pour les éléments retenus.

exemple5-24.php

```php
1 //Définition du tableau
2 $villes=array("Paimpol","Angers","Pau","Nantes","Lille");
3 //Fonction de sélection
4 function init($ville) {
5   if($ville[0]=="P" || $ville[0]=="p") return $ville;
6 }
7 //Utilisation de array_filter()
8 $select=array_filter($villes,"init"); print_r($select);
```

Transformation de tableaux

`bool array_walk(array &$tab, callable $f [, mixed $userdata]`

Applique la callback `$f` à chaque élément de `$tab`.

- bool `$f(mixed $v, mixed $c)` prend comme arguments la `v`aleur et la `c`lé de chaque élément. Accepte `$userdata` comme 3ème argument si défini.
- Ne peut modifier les valeurs de `$tab` qu'à la condition de spécifier `$v` comme référence :

  bool `$f(mixed &$v, mixed $c);`

exemple5-25.php

Manipulation du pointeur interne de tableaux

- `mixed current(array $tab)`
  - Retourne l'élément courant de `$tab`.
- `array each(array &$tab)`
  - Retourne valeur et clé de l'élément courant de `$tab`.

- `mixed end(array &$tab)`
  - Retourne la valeur du dernier élément de `$tab` (ou `FALSE` si `$tab` est vide) et positionne le pointeur en fin de `$tab`.
- `mixed next(array &$tab)`
  - Avance le pointeur interne de `$tab` et retourne l'élément courant ou `FALSE`.
- `mixed prev(array &$tab)`
  - Recule le pointeur interne de `$tab` et retourne l'élément courant ou `FALSE`.