LES BASES DU LANGAGE PHP

http://php.net/manual/fr/langref.php
http://www.w3schools.com/php/
https://openclassrooms.com/fr/courses/918836-concevez-votre-site-web-avec-php-et-mysql

SOMMAIRE

SOMMAIRE **1**

BASES du LANGAGE PHP **4**

0. Présentation du document **4**

Remarque 4
Document « slidé » 4
Objectifs généraux de ce document 5
Exemples 5
Exercices 6

1. Les variables **7**

Définition d'une variable 7
Les types 8
Vérifier le type 9
La signification 9
Que peut-on faire avec une variable ? 10
Opérateurs de base 11
exemple-1 : on peut utiliser une variable dans un calcul 11

2. Constantes **12**

Exemple-1.b : utilisation de constantes 12

3. Tests : exemple-s 2 et 3 ..... **13**

if, else, elseif 13
opérateurs de comparaison 13
opérateurs logiques 13
switch 14
opérateur de comparaison ternaire : ? : 14

4. HTML dans le PHP ou séparé du PHP : exemple-4 ..... **15**

HTML dans le PHP 15
HTML séparé du PHP 15
Alternative inutile : HTML dans le PHP avec balise heredoc : <<< 16

5. Boucles : exemple-5 ..... **17**

While 17
for 17
Exercice-1 : 10 premiers entiers, carrés et racines dans un tableau HTML 18

6. Débranchements **19**

Présentation 19
break 19
continue 19
goto 19

7. Bibliothèque de Fonctions **20**

Présentation 20
Fonctions de calcul mathématique 20
Fonction de traitement de chaine de caractères 20
Fonction de traitement de date 20
Envoi de mail 21
Générer des PDF en PHP 21

Générer des images en PHP 21
Traiter des expressions régulières en PHP 21
8. Tableau numéroté – exemple-s 6, 7 et 8 22
Présentation 22
Fonction array : créer le tableau 22
Créer le tableau par les indices 22
Créer le tableau par les indices automatiques 22
Accès à une valeur du tableu : par la clé 22
Instruction echo : afficher un élément du tableau 22
Boucle for : afficher tout le tableau 23
Attention : il ne doit pas y avoir de trous dans le tableau 23
boucle foreach : afficher tout le tableau, quel que soit l'indice 24
boucle foreach $key => $value 24
Fonction print-r() : affichage basique du tableau sans mise en forme 25
Fonction var_dump() : affichage basique des informations d'une variable 25
Chaine vers tableau : implode et explode 26
Exercice-2 tableau de prénoms et liste à puces 27
9. Tableau associatif – exemple-9 28
Présentation 28
fonction array : créer le tableau : 28
Créer le tableau par les indices 28
Accès à une valeur du tableu : par la clé 28
Instruction echo : afficher un élément du tableau 28
Afficher tout le tableau associatif : boucle foreach 29
boucle foreach $key => $value 29
Fonction print-r() : affichage basique du tableau associatif sans mise en forme 30
Exercice-3 un-user-Etape-0 31
Exercice-4 tableau périodique des éléments 31
10. Tableau numéroté de tableau associatif – exemple-10 32
11. Fonctions de manipulation de tableau – exemple-s 11 et 12 33
array 33
count - sizeof 33
sort – exemple-11 33
shuffle – exemple-11 33
array-search – exemple-11 33
ksort 34
asort 34
in_array 34
array_key_exists – exemple-12 34
Exercice-5 tableau-users-Etape-1 35
12. Écrire ses propres fonctions 36
Exemple d'affichage sans fonction– exemple-13 36
Fonction d'affichage, qui ne renvoie rien – exemple-13-a 37
Fonction qui renvoie un résultat – exemple-14 38
Fonction avec un paramètre en sortie : qui est modifié - & - exemple-15 39
13. Visibilité des variables – exemple-16 – global, GLOBALS, static 40
3 niveaux de visibilité pour les variables 40
Circulation de l'information 41
14. Exercice et dernières fonctionnalités sur les tableau : filtre et tri par colonne 42
Exercice-6 tableau-users-Etape-2 : codage avec fonctions 42
Filtrer un tableau : fonction array_filter – exemple-17 45
Exercice-7 tableau-users-Etape 3 : tri des données 46
exemple-18 : algorithme de tri récursif 47

Edition : sept 2021

BASES DU LANGAGE PHP

0. Présentation du document

- Les exemples sont présentés dans un chapitre en vert avec le mot clé : exemple-
- Les dossiers d'exemples sont fournis dans l'article qui contient ce fichier de cours.
  - ➔ http://bliaudet.free.fr/IMG/zip/PHP-01-Bases_du_langage.zip
  - ➔ Chargez ce fichier et mettez-le dans le dossier « php » du répertoire web « www » du serveur WEB. Vous pouvez aussi structurer les choses avec des dossiers J1, J2, etc. correspondant aux journées de travail.

- Les exercices à faire sont présentés dans un chapitre en jaune avec les mots-clés : TP- et exercice-
  - ➔ Les sources pour les exercices, quand il y en a, sont fournis dans le dossier des exemples.

Remarque

- Certains fichiers d'exemple commencent par ces trois lignes :

```
echo '<h1>CODE PHP</h1>';
highlight_file('fichier.php');
echo '<h1>RESULTATS</h1>';
```

- Ce code affiche deux balises h1 avec CODE PHP puis RESULTATS
- La fonction « highlight_file » permet d'afficher le contenu du fichier proposé. Quand on teste le code, on commence par affiche le code. Ca permet de voir le code en même temps que les résultats.
- Pour généraliser le code, on écrit : highlight_file(basename(__FILE__));
- basename(__FILE__) permet de récupérer le nom du fichier en cours de traitement.

Document « slidé »

- Ce document Word est en partie « slidé » : chaque page tient, en général, sur un écran, comme un slide.

Objectifs généraux de ce document

- Bases du fonctionnement du PHP : client-serveur, traduction HTML
- Environnement de développement : Laragon, WAMP, etc.
- Instructions echo et include
- Débogage

Exemples

- exemple-1 : variable dans un calcul
- Exemple-1.b : utilisation de constantes
- exemple-s 2 et 3 : tests
- exemple-4 : HTML dans le PHP ou séparé du PHP
- exemple-5 : Boucles
- exemples 6, 7 et 8 : Tableau numéroté
- exemple-9 : Tableau associatif
- exemple-10 : Tableau numéroté de tableau associatif
- exemples 11 et 12 : Fonctions de manipulation de tableau
- exemple-13 : Exemple d'affichage sans fonction
- exemple-13-a : Fonction d'affichage, qui ne renvoie rien
- exemple-14 : Fonction qui renvoie un résultat
- exemple-15 : Fonction avec un paramètre en sortie : qui est modifié - &
- exemple-16 : Visibilité des variables – global, GLOBALS, static
- exemple-17 : Filtrer un tableau : fonction array_filter
- exemple-18 : algorithme de tri récursif

Exercices

- Exercice-1 : 10 premiers entiers, carrés et racines dans un tableau HTML
  ➔ Objectif : boucle et affichage HTML dans le PHP
- Exercice-2 : tableau de prénoms et liste à puces
  ➔ Objectif : tableau numéroté et affichage HTML dans le PHP
- Exercice-3 : un-user-Etape-0
  ➔ Objectif : tableau associatif
- Exercice-4 : tableau périodique des éléments
  ➔ Objectif : tableau associatif
- Exercice-5 : tableau-users-Etape-1
  ➔ Objectif : tableau numéroté de tableaux associatifs (équivalent tableau d'objets)
- Exercice-6 : tableau-users-Etape-2 : codage avec fonctions
  ➔ Objectif : codage avec fonctions et regroupement des fonctions. Approche objet sans POO.
- Exercice-7 : tableau-users-Etape 3 : tri des données
  ➔ Objectif : utilisation de fonctions pour trier.

1. Les variables

Définition d'une variable

- Une variable est un moyen pour stocker en mémoire une information le temps de la génération de la page PHP
- **Une variable a :**
  - un nom
  - une valeur
  - un type
  - une signification.
- En PHP, le nom d'une variable commence par un $
- Par exemple : $username = « Bertrand »
  - Nom : username
  - valeur : « Bertrand »
  - type : String
  - signification : le nom d'un utilisateur

Les types

Présentation

- Les variables peuvent enregistrer des informations de différents types :
- Entier, décimal (nombre à virgule), texte et booléen (vrai ou faux) sont les principaux.
- http://php.net/manual/fr/language.types.php

Le type entier : int

- 0, 1, 2, 3, etc et les entiers négatifs : -1, -2, etc.

Le type décimal : float

- Ce sont les nombres à virgules : 1.234
- On écrit la virgule avec un point.
- Tant qu'on n'a pas besoin d'une précision extrême, ils conviennent très bien à tous les calculs.

Le type texte (ou chaine de caractères) : string

- Les chaines de caractères sont écrites entre guillemets ou apostrophe.
- On parle aussi de quotte ou simple quotte ou simple guillemet pour les apostrophes, de double quotte pour les guillemets.

Le type booléen : bool

- Peut prendre les valeurs true ou false.

La valeur NULL

On peut donner la valeur NULL à toutes les variables, quel que soit leur type.
Cela veut dire que la variable ne contient rien.

Types tableau, objet, ressource

Il existe aussi un type pour les tableaux (suite du cours), pour les objets (cours POO) et pour les ressource (une ressource est une référence à une ressource externe : voir la doc PHP).

Vérifier le type

- Quand on crée une variable, on n'a pas besoin de lui préciser son type.
- Selon la valeur qu'on donne à la variable, le type est défini automatiquement.
- Des fonctions permettent de savoir de quel type est quelle variable. Par exemple : is_bool($maVariable) retourne vrai (1) si $maVariable est un booléen, faux (0) sinon.
  ➔ http://php.net/manual/fr/function.is-bool.php
- is_bool, is_float, is_numeric, etc.

La signification

- Une variable sert à quelque chose, par exemple à enregistrer le nom de l'utilisateur.
- C'est sa signification. On lui donne un nom en rapport avec sa signification.
- Ce n'est pas obligé, mais c'est préférable.

Que peut-on faire avec une variable ?

On peut donner une valeur à une variable

```
$username= "Barack";
```

On parle d'affectation ou d'assignation

On peut afficher le contenu d'une variable

```
echo $username;
```

Ici pas de guillemets comme quand on affiche un texte.

On peut concaténer une variable à une chaine de caractère

Avec du texte entre apostrophes ou entre guillemets, on peut concaténer une variable avec l'opérateur « . »

```
Echo 'bonjour' . $username. '. Comment allez-vous ?'
```

Mieux vaut utiliser les apostrophes « ' » dans le PHP et les guillemets « " » dans le HTML.

On peut afficher le contenu d'une variable dans une chaine de caractère

Avec du texte entre guillemets, on peut mettre la variable directement dans le texte entre guillemets.

```
Echo "bonjour $username. Comment allez-vous ?"
```

On évite cet usage. On utilise le précédent, avec la concaténation

Opérateurs de base

Concaténation : opérateur « . »

Le « . » permet de concaténer deux textes, une variable et du texte, deux variables.

Opérateurs arithmétique

On utilise les opérateurs classiques :
+, -, /, *, %

Opérateur d'incrémentation

$i++ équivaut à $i=$i+1
Attention, c'est une post-incrémentation : si on fait « echo $i++ ; c'est le $i avant l'incrémentation qui est affiché.

exemple-1 : on peut utiliser une variable dans un calcul

```
<?php
   $prix_unitaire=11.6;
   $quantite=5;
   $produit="clé USB 32 GO";
   $prix_total=$quantite*$prix_unitaire;
   echo 'bonjour <br>';
   echo $quantite. ' ' .$produit. ' : ' .$prix_total;
?>
```

On peut utiliser les symboles classiques de calcul : +, - , x, /, % (modulo), et toutes les fonction classiques (sin(), cos(), sqrt(), pow() : http://php.net/manual/fr/function.pow.php
A noter que à gauche du signe =, on modifie la valeur de la variable. Ce qu'il y avait dans la variable avant la modification est perdu. A droite du signe =, on utilise la valeur de la variable pour faire le calcul.
A noter aussi qu'on ne met pas de point autour de la variable dans l'echo, au début de l'echo, au début de la variable, à la fin de la variable à la fin de l'echo.

2. Constantes

- Une constante est une sorte de variable qui ne peut pas être modifiée.
- Son nom est donné en majuscule par convention.
- Sa valeur est donnée par la fonction « define ».
- On y accède sans utiliser le « $ ».

Exemple-1.b : utilisation de constantes

```php
<?php
define("CONSTANT", "Bonjour le monde.");
echo CONSTANT; // affiche "Bonjour le monde."

define('ANIMALS', array(
    'chien',
    'chat',
    'oiseaux'
));
echo ANIMALS[1]; // affiche "chat"

define('ANIMAL', array(
    'chien' => 'Droopy',
    'chat' => 'Felix',
    'canari' => 'Titi',
));
echo ANIMAL['chat']; // affiche "Felix"

?>
```

➔ On utilise un tableau et un tableau associatif. On verra dans la suite de ce cours ces deux types de tableau.

3. Tests : exemple-s 2 et 3

if, else, elseif

```
if ($maVariable == 0){
   instructions ;
}
elseif ($maVariable >0 {
   instructions ;
}
else {  /* $maVariable<0 */
   instructions ;
}
```

if : http://php.net/manual/fr/control-structures.if.php
else : http://php.net/manual/fr/control-structures.else.php
elseif : http://php.net/manual/fr/control-structures.elseif.php

opérateurs de comparaison

==, !=, <, <=, >, >=
http://php.net/manual/fr/language.operators.comparison.php#language.operators.comparison
=== : vrai si les valeurs sont identiques et de même type, !==

opérateurs logiques

binaires : AND, &&, OR, || ($a>0 and $a<10)
unaires : ! (!is_int($a))
http://php.net/manual/fr/language.operators.logical.php

switch

```
switch ($maVariable )
{
  case(0) :
    instructions ;
    break ;
  case (1) :
    instructions ;
    break ;
  default :
    instructions ;
}
```

switch : http://php.net/manual/fr/control-structures.switch.php

opérateur de comparaison ternaire : ? :

$monResultat = ($maVarialble == 0) ? true : false ;
http://php.net/manual/fr/language.operators.comparison.php#language.operators.comparison.ternary

4. HTML dans le PHP ou séparé du PHP : exemple-4

HTML dans le PHP

- Cette solution réunit le code HTML et le code PHP.
- On peut mettre les balises HTML dans l'echo et les variables dans l'echo.
- Il faut utiliser des \' ou des guillemets pour gérer les apostrophes dans le texte.

```php
<?php
  if ($variable <0 ) {
    echo '<strong> La valeur : </strong>'.$variable.' est négative ! <br>';
    echo 'C\'est une façon de faire<br>';
    echo "C'en est une autre";
  }
?>
```

➔ La coloration syntaxique n'est pas pratique : on ne voit pas le HTML.

➔ C'est verbeux : il faut ajouter des echo

HTML séparé du PHP

- Cette solution sépare le code HTML du code PHP.

```php
<?php if ($variable <0 ) { ?>
  <strong>La valeur : </strong> <?php echo $variable ?> est négative !
  C'est une façon de faire<br>
  C'en est une autre
<?php } ?>
```

➔ C'est plus adapté car la coloration syntaxique est plus claire.

➔ De plus, ça se rapproche des langages de template comme Blade de Laravel.

- On n'a plus qu'un seul bloc : le bloc « if »
- Et trois lignes qui correspondent à 3 lignes HTML (ça pourrait être une balise <p> à chaque fois).

Alternative inutile : HTML dans le PHP avec balise heredoc : <<<

- L'opérateur <<< qu'on appele « balise heredoc », permet d'ouvrir une une balise avec le nom qu'on veut : ici _BALISE
- Tout le texte contenu dans la balise est une chaîne de caractère.
- On peut y mettre directement des variables $variable

```php
<?php
  if ($variable <0 ) {
      echo <<<_BALISE
          <strong>La valeur : </strong> $variable est négative ! <br>
          C'est une façon de faire <br>
          C'en est une autre
      _BALISE;
  }
?>
```

➔ La coloration syntaxique n'est pas pratique : on ne voit pas le HTML.

➔ C'est moins verbeux : on n'a plus les echos

➔ Ce n'est pas utilisé !!!

5. Boucles : exemple-5

While

- http://php.net/manual/fr/control-structures.while.php

Présentation

```
while ($cpt < 10){
  instructions ;
}
```

- tant que maVariable <10, on répète les instructions.
- Il faut bien sûr que maVariable soit modifiée dans les instructions pur qu'on puisse sortir de la boucle.

Boucler 10 fois

- Pour boucler 10 fois on peut écrire

```
$cpt =1
while ($cpt <= 10){
  instructions ;
  $cpt ++ ;
}
```

- Le ++ permet d'incrémenter $cpt.
- $cpt ++ ; est équivalent à $cpt =$cpt +1 ;

for

http://php.net/manual/fr/control-structures.for.php

- La boucle for est l'équivalent de la boucle while pour boucler 10 fois (ou autant de fois qu'on veut) :

```
for ($cpt =1 ; $cpt < 1 ; $cpt++){
          instructions ;
}
```

- Il y a trois parties dans le for :
- L'initialisation de maVariable est dans le for, en premier.
- La condition de sortie est au milieu.
- L'incrémentation de maVariable est en troisième position.

Exercice-1 : 10 premiers entiers, carrés et racines dans un tableau HTML

- Ecrire un script qui affiche les 10 premiers entiers, le carré et leur racine carré dans un tableau.
- Chercher la fonction racine carré.
- Résultat attendu :

Nombre	Carré	Racine
1	1	1
2	4	1.41421356237
3	9	1.73205080757
4	16	2

etc.

- On fera une version avec 2 chiffres après la virgules pour la racine.

6. Débranchements

Présentation

- 2 instructions permettent de quitter le fonctionnement standard des boucles : break qui fait quitter la boucle et continue qui permet de passer au suivant.

break

- http://php.net/manual/fr/control-structures.break.php

```
while ($maVariable < 10){
  if($casParticulierQuiFaitSortir ==true){
            ce qu'il y a à faire
            break; //
          }
          cas général
}
```

continue

- http://php.net/manual/fr/control-structures.continue.php

```
while ($maVariable < 10){
  if($casParticulierQuiFaitPasserAuSuivant ==true){
            ce qu'il y a à faire
            continue; // (on passe au suivant)
          }
          cas général
}
```

goto

- http://php.net/manual/fr/control-structures.goto.php
- Le goto permet d'aller de n'importe où vers n'importe où !
- C'est à éviter !

7. Bibliothèque de Fonctions

Présentation

Il existe des milliers de fonctions qu'on peut utiliser en PHP.
http://php.net/manual/fr/funcref.php

Fonctions de calcul mathématique

http://php.net/manual/fr/book.math.php

- sqrt : racine carré, pow : puissance, round : arrondi, rand : aléatoire,
- min, max, cos, sin, etc.

Fonction de traitement de chaine de caractères

http://php.net/manual/fr/ref.strings.php

- length, substr, strpos, str_replace : longueur, extraction, position, remplacement.
- trim, ltrim : pour supprimer les espaces au début ou à la fin d'une chaîne.
- chr, ord : passer d'un caractère à un nombre
- printf, fprintf : print formaté (f de fin), print formaté dans un fichier (f de début)
- sscanf : scan formaté (f de fin) dans une string (s de début)

Fonction de traitement de date

http://php.net/manual/fr/ref.datetime.php

- getdate() pour récupérer la date et heure du jour :
  https://www.php.net/manual/fr/function.getdate.php

Envoi de mail

- http://php.net/manual/fr/function.mail.php

Générer des PDF en PHP

- http://php.net/manual/fr/book.pdf.php

Générer des images en PHP

- http://php.net/manual/fr/book.image.php
- OCR

Traiter des expressions régulières en PHP

Généralités sur les expressions régulières

- https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql/memento-des-expressions-regulieres

Expression régulière en PHP

- http://php.net/manual/fr/book.pcre.php
- OCR_1 et OCR_2

8. Tableau numéroté – exemple-s 6, 7 et 8

Présentation

- Tableau « classique » : permet de mettre plusieurs valeurs d'un même type dans une même variable.

Fonction array : créer le tableau

- $prenoms = array ('Aurélien', 'Isabelle', 'Ahmed', 'Olivier' , 'Nour', 'Chang') ;

// crée $prenoms[0], $prenoms[1] jusqu'à $prenoms[5]

Créer le tableau par les indices

```
$lesPrenoms[0]=' Aurélien' ;
$lesPrenoms[1]=' Isabelle';
$lesPrenoms[2]=' Ahmed' ;
```

➔ Notez que le premier élément du tableau est à l'indice 0.

➔ L'indice est aussi appelé clé.

Créer le tableau par les indices automatiques

```
$lesPrenoms [ ]=' Aurélien' ; // crée $prenoms[0]
$lesPrenoms [ ]=' Isabelle';  // crée $prenoms[1]
$lesPrenoms [ ]=' Ahmed' ;    // crée $prenoms[2]
```

➔ Quand on utilise les crochets vides, le nouvel élément est placé automatiquement à la suite des précédents.

Accès à une valeur du tableu : par la clé

- Pour accéder à un élément du tableau on écrit : $tableau[indice]
- Par exemple : $lesPrenoms[1] ;

Instruction echo : afficher un élément du tableau

```
echo $prenoms[2] ;
```

Boucle for : afficher tout le tableau

```php
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';

   for ($i = 0 ; $i <count($lesPrenoms) ; $i++) {
      echo 'lesPrenoms['.$i.'] = ' .$lesPrenoms[$i].  '<br/>' ;
   }
?>
```

➔ Notez la présence de la fonction count pour avoir le nombre d'éléments du tableau.

Attention : il ne doit pas y avoir de trous dans le tableau

```php
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   // le for n'affichera rien pour $lesPrenoms[3]
   // et  n'affichera pas $lesPrenoms[4]
   for ($i=0 ; $i <count($lesPrenoms) ; $i++) {
      echo 'lesPrenoms['.$i.'] = ' .$lesPrenoms[$i].  '<br/>' ;
   }
?>
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   echo '<h2>affichage foreach $key => $value</h2>';
   foreach ($lesPrenoms as $key => $value){
      echo '$key : '.$key. ' => ' ;
      echo '$value : '.$value. '<br/>' ;
   }
?>
```

➔ **Notice**: Undefined offset: 3 in **C:\laragon\www\php\test.php** on line **10**
lesPrenoms[3] =

boucle foreach : afficher tout le tableau, quel que soit l'indice

- Pour passer en revue tous les éléments du tableau, quel que soit leur numéro dans le tableau, on utilise le foreach.
- La boucle foreach gère automatiquement le fait de démarrer au premier élément du tableau et d'aller jusqu'au dernier en sautant les trous.
- Dans les paramètres du foreach, on précise le tableau et le nom de chaque élément qu'on va récupérer, séparés par un « as ».

```php
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   foreach ($lesPrenoms as $unPrenom){
      echo $unPrenom. '<br/>' ;
   }
?>
```

boucle foreach $key => $value

- Chaque élément d'un tableau correspond à un couple (key, value) :
  - ➔ La clé est le numéro de l'élément dans le tableau, la value sa valeur.
  - ➔ On écrit : $key => $value

```php
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Bérénice';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   echo '<h2>affichage foreach $key => $value</h2>';
   foreach ($lesPrenoms as $key => $value){
      echo '$key : '.$key. ' => ' ;
      echo '$value : '.$value. '<br/>' ;
   }
?>
```

  - ➔ $key : 0 => $value : Aurélien
  - ➔ $key : 1 => $value : Bérénice
  - ➔ Etc.

Fonction print-r() : affichage basique du tableau sans mise en forme

- Affichage sur une seule ligne et en une seule ligne :

```
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   print_r($lesPrenoms);
?>
```

➔ Array ([0] => Aurélien [1] => Olivier [2] => Ahmed [5] => Hang)

- Affichage ligne par ligne : avec la balise <pre>

```
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   echo '<pre>'; print_r($lesPrenoms); echo '</pre>';
?>
```

➔ Array
(
[0] => Aurélien
[1] => Olivier
[2] => Ahmed
[5] => Hang
)

Fonction var_dump() : affichage basique des informations d'une variable

- Affichage sur une seule ligne avec le type en plus :

```
<?php
   $lesPrenoms[0]='Aurélien';
   $lesPrenoms[1]='Olivier';
   $lesPrenoms[2]='Ahmed';
   $lesPrenoms[5]='Hang';

   var_dump($lesPrenoms);
   $i=5;
   var_dump($i);
?>
```

➔ array(4) { [0]=> string(9) "Aurélien" [1]=> string(7) "Olivier" [2]=> string(5) "Ahmed" [5]=> string(4) "Hang" } int(5)

Chaine vers tableau : implode et explode

Présentation

- explode() : pour convertir une chaine avec une liste de valeurs en tableau numéroté.
  - ➔ https://www.php.net/manual/en/function.explode.php
  - ➔ https://www.w3schools.com/php/func_string_explode.asp

- Implode() : pour convertir un tableau numéroté en une chaine contenant la liste des valeurs du tableau.
  - ➔ https://www.php.net/manual/en/function.implode.php
  - ➔ https://www.w3schools.com/php/func_string_implode.asp

- On précise aux fonction le séparateur qu'on trouve entre les valeurs dans la chaine.

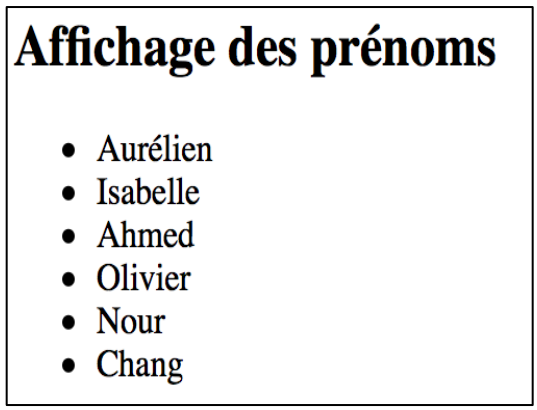
Exemples

- $chaine=implode ($sep, $tableau) ;
- $chaine=implode («
 », $tableau) ;

- $tab = explode ($sep, $chaine) ;
- $tab = explode (« , », $chaine) ;

Exercice-2 tableau de prénoms et liste à puces

- Ecrire un script qui affiche le contenu d'un tableau de prénoms dans une liste à puces. Les prénoms sont écrits en minuscules avec une majuscule en premier.
- Résultat attendu :



- Faites une version qui n'affiche que les prénoms dont la première lettre vient après le H.
- Chercher des informations sur la fonction strcmp() dans la documentation PHP.
- On utilisera un « continue ».

9. Tableau associatif – exemple-9

Présentation

- Un tableau associatif est un tableau particulier qui permet de mettre plusieurs valeurs de types différents dans une même variable.
- C'est **l'équivalent d'un objet** ou d'une structure dans d'autres langages.
- **A chaque valeur on associe un nom qu'on appelle clé ou attribut**. Cette clé est donc « associé » à la valeur. PHP parle de tableau associatif.

fonction array : créer le tableau :

```
$utilisateur = array (
  'nom' => 'Toto',
  'prenom' => 'Aurélien',
  'dateNaissance' => 1995,
  'nomUtilisateur' => 'aurelien1995'
) ;
```

➔ 'nom' est une clé qui permet d'accéder à la valeur 'Toto'

Créer le tableau par les indices

```
$utilisateur['nom'] = 'Toto';
$utilisateur['prenom'] = 'Aurélien';
$utilisateur['dateNaissance'] = 1995;
$utilisateur['nomDUtilisateur'] = 'aurelien1995';
```

➔ 'nom' est une clé qui permet d'accéder à la valeur 'Toto'

Accès à une valeur du tableu : par la clé

- Pour accéder à un élément du tableau on écrit : $tableau[‘clé’]
- Par exemple : $utilisateur[‘nomUtilisateur’] ;

Instruction echo : afficher un élément du tableau

```
echo $utilisateur['nomDUtilisateur']
```

Afficher tout le tableau associatif : boucle foreach

- Pour passer en revue tous les attributs du tableau associatif, on utilise le foreach.
- Dans les paramètres du foreach, on précise le tableau et le nom de chaque élément qu'on va récupérer, séparés par un « as ».

```php
<?php
   $utilisateur = array (
      'nom' => 'Toto',
      'prenom' => 'Aurélien',
      'dateNaissance' => 1995,
      'nomUtilisateur' => 'aurelien1995'
   );

   foreach ($utilisateur as $value){
      echo $value. '<br/>';
   }
?>
```

boucle foreach $key => $value

- Chaque élément d'un tableau correspond à un couple (key, value) :
  - ➔ La clé est le nom de l'attribut, la value sa valeur.
  - ➔ On écrit : $key => $value

```php
<?php
   $utilisateur = array (
      'nom' => 'Toto',
      'prenom' => 'Aurélien',
      'dateNaissance' => 1995,
      'nomUtilisateur' => 'aurelien1995'
   );

   echo '<h2>affichage foreach as $key => $value</h2>';
   foreach ($utilisateur as $key => $value){
      echo '$key: '.$key. ' => ' ;
      echo '$value: '.$value. '<br/>' ;
   }
?>
```

  - ➔ $key: nom => $value: Toto
  - ➔ $key: prenom => $value: Aurélien
  - ➔ Etc.

Fonction print-r() : affichage basique du tableau associatif sans mise en forme

- Affichage sur une seule ligne

```php
<?php
   $utilisateur = array (
      'nom' => 'Toto',
      'prenom' => 'Aurélien',
      'dateNaissance' => 1995,
      'nomUtilisateur' => 'aurelien1995'
   );

   print_r($utilisateur);
?>
```

➔ Array ([nom] => Toto [prenom] => Aurélien [dateNaissance] => 1995 [nomUtilisateur] => aurelien1995)

- Affichage ligne par ligne : avec la balise <pre>

```php
<?php
   $utilisateur = array (
      'nom' => 'Toto',
      'prenom' => 'Aurélien',
      'dateNaissance' => 1995,
      'nomUtilisateur' => 'aurelien1995'
   );

   echo '<pre>'; print_r($utilisateur); echo '</pre>';
?>
```

➔ Array
(
[nom] => Toto
[prenom] => Aurélien
[dateNaissance] => 1995
[nomUtilisateur] => aurelien1995
)

Exercice-3 un-user-Etape-0

- On veut gérer des utilisateurs avec les caractéristiques suivantes :
  - Prénom et NOM (dans un seul champ),
  - mail,
  - motDePasse,
  - age

- Créer un utilisateur (vous !) avec ces informations dans un tableau associatif.
- Afficher ce tableau associatif avec un print_r avec une ligne par information.
- Afficher ce tableau associatif dans un tableau HTML.

Exercice-4 tableau périodique des éléments

- En chimie, le tableau périodique des éléments associe un symbole à un nom d'élément chimique :
  - H pour Hydrogène,
  - He pour Helium,
  - etc.
- Faites un programme qui affiche au moins les 5 premiers éléments dans un tableau HTML.
- Vous pouvez trouver les autres sur internet.

- Résultats attendus : vous devez mettre les résultats dans une page HTML.

Symbole	Element
H	Hydrogene
He	Helium
Li	Lithium
Be	Beryllium
B	Bore

10. Tableau numéroté de tableau associatif – exemple-10

- On utilise souvent des tableaux numérotés qui contiennent des tableaux associatifs.
- Par exemple, un tableau d'élèves.
- On utilise un foreach et un foreach $key=>$value.

```php
<?php
   // ecriture 1
   $utilisateurs[0] = array ( // c'est $utilisateurs[0]
      'nom' => 'Toto',
      'prenom' => 'Lolo',
      'dateNaissance' => 1995,
      'nomUtilisateur' => 'toto1995'
   ) ;

   // ecriture 2
   $utilisateurs[] = array ( // c'est $utilisateurs[1]
      'nom' => 'Tata',
      'prenom' => 'Lala',
      'dateNaissance' => 1992,
      'nomUtilisateur' => 'tata1992'
   ) ;

   // ecriture 3
   $utilisateurs[2]['nom'] ='Titi';
   $utilisateurs[2]['prenom'] ='Lili';
   $utilisateurs[2]['dateNaissance'] ='1990';
   $utilisateurs[2]['nomUtilisateur'] ='titi1990';

   // ecriture 4 - pas de tableau asso - les clés seront des indices
   $utilisateurs[3] = ['Titi4', 'Ahmed', 1990, 'ahmed1990'] ;

   echo '<h2>affichage print_r</h2>';
   print_r($utilisateurs);

   echo '<h2>affichage foreach</h2>';
   foreach ($utilisateurs as $user){
      echo '<pre>'; print_r($user); echo '</pre>';
   }

    echo '<h2>affichage foreach sindex => $user</h2>';
    foreach ($utilisateurs as $index => $user){
      echo('----------------------------------------<br>');
      echo '<b>$index '.$index. ' : </b><br>';
      foreach ($user as $key => $value){
           echo '$key : '.$key. ' => ';
           echo '$value : '.$value. '<br>';
       }
    }

?>
```

11. Fonctions de manipulation de tableau – exemple-s 11 et 12

- Il existe beaucoup de fonction de manipulation de tableau
  - ➔ http://php.net/manual/fr/ref.array.php
- Citons particulièrement :

array

➔ pour créer un tableau

count - sizeof

➔ count($monTableau) ou sizeof($monTableau) pour récupérer le nombre d'éléments du tableau.

sort – exemple-11

➔ sort($monTableau) pour trier un tableau numéroté.

➔ rsort() pour trier en sens inverse.

➔ Toutes les fonctions de tri : http://php.net/manual/fr/array.sorting.php

shuffle – exemple-11

➔ shuffle($monTableau) pour mélanger un tableau.

array-search – exemple-11

➔ array_search('Isabelle', $lesPrenoms) renvoie l'indice de la valeur 'Isabelle' dans le tableau $lesPrenoms.

ksort

- ksort($monTableau) pour trier un tableau associatif selon la « key ».
- krsort() pour trier en sens inverse.

asort

- asort($monTableau) pour trier un tableau associatif selon la « value». C'est équivalent à sort() pour un tableau numérotée.
- arsort() pour trier en sens inverse. C'est équivalent à rsort() pour un tableau numéroté.

in_array

- in_array('valeur', $monTableau) est vrai si 'valeur' est dans $monTableau.

array_key_exists – exemple-12

- array_key_exists($cle, $monTableauAssociatif) est vrai si $cle est une clé du tableau associatif $monTableauAssociatif.

Exercice-5 tableau-users-Etape-1

1. On veut gérer des utilisateurs avec les caractéristiques suivantes :
   - Prénom et NOM (dans un seul champ), mail, motDePasse, age
     - ➔ Créer un utilisateur (vous !) avec ces informations dans un tableau associatif.
     - ➔ Afficher ce tableau associatif avec un print_r avec une ligne par information.

2. On veut créer non plus un seul utilisateur mais un tableau d'utilisateurs.
   - Créer ce tableau avec 5 utilisateurs.
   - On créera le premier utilisateur avec la syntaxe suivante :

```
$lesUtilisateurs[0]['nom']='Toto TOTO';
etc.
```

   - Et les autres avec cette syntaxe :

```
$lesUtilisateurs[] = array(
    'nom'=>'Toto TOTO',
    etc.
```

     - ➔ Afficher les utilisateurs créés avec un print_r
     - ➔ Afficher les utilisateurs avec un for
     - ➔ Afficher les utilisateurs avec un foreach
     - ➔ Afficher le tableau des 5 utilisateurs en HTML dans un tableau HTML.

12. Écrire ses propres fonctions

Exemple d'affichage sans fonction– exemple-13

- On veut afficher « Lolo a 25 ans » à partir d'un utilisateur avec son nom, son préom, son année de naissance et son nom d'utilisateur.

- On structure le code en format MVC
  - D'abord le modèle : on définit les données : l'utilisateur
  - Ensuite le Controleur : on fait les calcul : ici l'âge à partir de la date du jour.
  - Enfin la vue : on affiche le résultat souhaité.

```php
<?php
    echo '<h1>CODE PHP</h1>';
    highlight_file(basename(__FILE__));
    echo '<h1>RESULTATS</h1>';

// Modèle
    $utilisateur = array (
        'nom' => 'Toto',
        'prenom' => 'Lolo',
        'anneeNaissance' => 1995,
        'nomUtilisateur' => 'lolo1995'
    );

// Contrôleur
    $today=getdate();
    $age=$today['year']-$utilisateur['anneeNaissance'];

// Vue
    echo '<h2>utilsation de getDate sans fonction</h2>';
    echo $utilisateur['prenom']. ' a ' .$age. ' ans';
?>
```

Fonction d'affichage, qui ne renvoie rien – exemple-13-a

- On va définir une fonction qui fait le calcul et l'affichage.
  - La fonction a deux paramètres en entrée : le prénom et l'année de naissance.
  - Elle calcule l'âge et affiche le résultat souhaité.

- On structure le code en format MVC
  - D'abord le modèle : on définit les données : l'utilisateur
  - Ensuite le Controleur : on définit la fonction d'affichage
  - Enfin la vue : on appelle la fonction d'affichage

```php
<?php
    echo '<h1>CODE PHP</h1>';
    highlight_file(basename(__FILE__));
    echo '<h1>RESULTATS</h1>';

// Modèle
    $utilisateur = array (
        'nom' => 'Toto',
        'prenom' => 'Lolo',
        'anneeNaissance' => 1995,
        'nomUtilisateur' => 'lolo1995'
    );

// Contrôleur
    // fonction d'affichage du prénom et de l'âge à partir
    //     du prénom et le l'année de naissance
    // E : prénom, année de naissance
    // S : affichage du résultat
    // Return : rien
    function afficherAge($prenom, $anneeNaissance){
        $today=getdate();
        $age=$today['year']-$anneeNaissance;
        echo $prenom. ' a ' .$age. ' ans' ;
    }

// Vue
    echo '<h2>utilsation de getDate avec fonction</h2>';
    afficherAge($utilisateur['prenom'], $utilisateur['anneeNaissance']);
?>
```

Fonction qui renvoie un résultat – exemple-14

- On reprend l'exemple précédent et on écrit une fonction qui retourne l'âge à partir de l'année de naissance.
  - La fonction a un seul paramètre en entrée : l'année de naissance.
  - Elle calcule l'âge et le retourne.

- On structure le code en format MVC
  - D'abord le modèle : on définit les données : l'utilisateur
  - Ensuite le Contrôleur : on définit la fonction de calcul. Ensuite on fait les calculs : on récupère le prénom et l'age.
  - Enfin la vue : on affiche le résultat souhaité à l'aide des variables calculées dans le contrôleur.

```php
<?php
    echo '<h1>CODE PHP</h1>';
    highlight_file(basename(__FILE__));
    echo '<h1>RESULTATS</h1>';

// Modèle
    $utilisateur = array (
        'nom' => 'Toto',
        'prenom' => 'Lolo',
        'anneeNaissance' => 1995,
        'nomUtilisateur' => 'lolo1995'
    );

// Contrôleur
    // les fonctions

    // fonction de calcul de l'âge avec l'année de naissance
    // E : année de naissance
    // S : rien
    // Return : age
    function calculerAge($anneeNaissance){
        $today=getdate();
        $year=$today['year'];
        $age=$year-$anneeNaissance;
        return $age;
    }

    // le calcul
    $prenom=$utilisateur['prenom'];
    $age=calculerAge($utilisateur['anneeNaissance']);


// Vue
    echo '<h2>utilsation de getDate avec fonction</h2>';
    echo $prenom. ' a ' .$age. ' ans';
?>
```

Fonction avec un paramètre en sortie : qui est modifié - & - exemple-15

- On va écrire une fonction qui augmente un employé.
- L'employé est un tableau associatif avec son nom et son salaire.
- La fonction reçoit en paramètre l'employé et l'augmentation.

```php
<?php
    echo '<h1>CODE PHP</h1>';
    highlight_file(basename(__FILE__));
    echo '<h1>RESULTATS</h1>';

    // fonction d'augmentation d'un employé
    // E : employé, augmentation
    // S : employé (on modifie son salaire)
    function augmenter(&$employe, $augmentation){
        $employe['salaire']= $employe['salaire']+$augmentation;
    }

    $emp=array('nom'=>'Toto', 'salaire'=>2000);
    echo '<h3>Avant augmentation : </h3>';
    print_r($emp);
    augmenter($emp, 200);
    echo '<h3>Après augmentation : </h3>';
    print_r($emp);
?>
```

➔ L'employé qui est un paramètre en entrée de la fonction, est modifiée : il faut donc le passer « par adresse ».

➔ Pour passer un paramètre par adresse en PHP, on met un « & » avant le paramètre dans la définition de la fonction.

➔ Si on ne met pas le « & », la modification n'est pas prise en compte : le salaire reste à 2000.

13. Visibilité des variables – exemple-16 – global, GLOBALS, static

3 niveaux de visibilité pour les variables

http://php.net/manual/fr/language.variables.scope.php

- **Les variables de la page ou variables globales** : elles sont visibles dans la page après leur première apparition, sauf dans les fonctions de la page, sauf si elles sont redéclarées « global » dans les fonctions. Elles ne sont pas visibles dans les autres pages.
  - A noter que les variables de page sont appelées en général variables globales (globales à la page).
  - A noter aussi qu'on peut utiliser le tableau associatif $GLOBALS qui contient les couples key-value correspondant aux couples nomDeVariable-valeurDeVariable.
- **Les variables locales** : elles ne sont visibles que dans la fonction où elles sont définies.
- **Les variables « static »** ou encore « locales-globales » : elles ne sont visibles que dans la fonction ou elles sont définies mais elles gardent leur valeur quand on revient dans la fonction.

Circulation de l'information

Circulation entre fonctions et entre page et fonctions

- Pour passer de l'information à une fonction, on la passe en paramètre de la fonction.
- Si une variable veut accéder à une variable de la page, elle doit la déclarer « global » dans la fonction.

```
<?php
   function test1() {
      $varGlobale='test1';
   }

   function test2() {
      global $varGlobale;
      $varGlobale='test2';
   }

   $varGlobale='global';
   test1();
   echo $varGlobale.'<br>'; // Affiche : global
   test2();
   echo $varGlobale.'<br>'; // Affiche : test2
?>
```

Circulation entre pages

- Pour faire circuler de l'information entre pages, on utilise les variables $_GET et $_POST et aussi la variable $_SESSION.
- Cf. chapitre suivant.

14. Exercice et dernières fonctionnalités sur les tableau : filtre et tri par colonne

Exercice-6 tableau-users-Etape-2 : codage avec fonctions

- On prend le fichier de correction de tableau-users-Etape-1 (exercice du chapitre 11).
- Il est accessible ici : http://bliaudet.free.fr/IMG/zip/tableau-users-Etape-1-correction.zip
- Dupliquer le dossier et appeler le tableau-users-Etape-2.php

1. Dans un nouveau fichier appelé « unUser.php » on va créer une fonction permettant de créer un utilisateur. On l'appelle « newUser ». Écrivez cette fonction.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// creation d'un utilisateur
newUser( parametres à déterminer);
```

2. Dans le fichier « unUser.php », ajoutez une fonction qui permette d'afficher un utilisateur avec un print_r.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// affichage du tableau avec un print_r
print_rUnUser( parametres à déterminer);
```

3. Dans le fichier « unUser.php », ajoutez une fonction qui permette d'afficher un utilisateur champ par champ.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// affichage du tableau avec un print_r
printUnUser( parametres à déterminer);
```

4. Dans un nouveau fichier appelé « lesUsers.php », ajoutez une fonction qui permette de créer un tableau d'utilisateurs. Les utilisateurs sont fournis dans la fonction. On appelle cette fonction initTab.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// initialisation d'un tableau d'utilisateurs
initLesUsers( parametres à déterminer );
```

5. Ensuite on se dote d'une fonction qui affiche les utilisateurs avec un print_r.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// affichage du tableau avec un print_r
print_rLesUsers ($lesUtilisateurs);
```

6. Ensuite on se dote d'une fonction qui affiche avec un for.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
// affichage du tableau de façon basique avec un for
print_rLesUsersFor ($lesUtilisateurs);
```

7. Ensuite on se dote d'une fonction qui affiche avec un foreach.
   ➔ Mettez à jour tableau-users-Etape-2.php pour qu'il utilise cette fonction.

```
            // affichage du tableau de façon basique avec un
foreach
            print_rLesUsersForeach($lesUtilisateurs);
```

8. **A la place des âges, on entre l'année de naissance**. Adapter le programme pour qu'on continue à afficher les âges. On se dote d'une fonction calculerAge($anneeNaissance) et et met à jour la fonction printUnUser(), le tout dans le fichier unUser.php.

9. Créer une fonction d'affichage qui soit stylé : par exemple on affiche dans un **tableau HTML et on gère un peu de CSS** pour le tableau.

Filtrer un tableau : fonction array_filter – exemple-17

- La fonction « array_filter » permet de filtrer un tableau.
- On lui passe le tableau à filtrer en paramètre ainsi qu'une fonction de filtre qu'on peut écrire comme on veut.
  - ➔ Cette fonction de filtre a en paramètre un élément du tableau et retourne true si on veut conserver l'élément, false si on ne veut pas le conserver.
  - ➔ Dans notre exemple, on garde les nombres pairs.

```php
<?php
    echo '<h1>CODE PHP</h1>';
    highlight_file(basename(__FILE__));
    echo '<h1>RESULTATS</h1>';

    $tab=array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

    echo '<h2>affichage tableau complet</h2>';
    echo'<pre>';print_r($tab);echo'</pre>';

    function filtreLesPairs($element){
        if($element%2==0) return true;
        else return false;
    }

    $tab_filtre=array_filter($tab, "filtreLesPairs");
    echo '<h2>affichage des pairs</h2>';
    echo'<pre>';print_r($tab_filtre);echo'</pre>';
?>
```

Exercice-7 tableau-users-Etape 3 : tri des données

- On veut trier les utilisateurs par nom.
  - ➔ On utilise la fonction « array_multisort ».
  - ➔ Son principe est de créer un tableau avec uniquement les noms, puis de faire appel à la fonction « array_multisort » en passant en paramètre le tableau avec les noms, une constante (SORT_ASC pour dire que c'est un tri croissant) et enfin le tableau des utilisateurs.
  - ➔ https://www.php.net/manual/fr/function.array-multisort

- On écrit une fonction qui fait le tri par nom avec un SORT_DESC
  - ➔ On passe les Users en paramètre

```
// fonction de tri décroissant par nom des utilisateurs
// E : le tableau $lesUsers
// S : le tableau $lesUsers trié
// Return : rien
function triParNomDesc(&$lesUsers){
   // tri du tableau par nom

   // on commence par fabriquer le tableau des noms
   foreach ($lesUsers as $indice => $unUser){
      $lesNoms[$indice]  = $unUser['nom'];
   }

   // appel à array_multisort : le tableau à trier est en dernier
   array_multisort($lesNoms, SORT_DESC, $lesUsers);
}
```

- Enregistrer cette fonction dans un fichier « tri.php » et tester cette fonction à partir des résultats de l'Etape 2.

- Avec la même méthode, trier par âge décroissant

- Avec la même méthode, trier par âge et nom décroissant

- Ensuite on va généraliser la fonction de tri pour qu'elle puisse trier selon n'importe quel champ.

exemple-18 : algorithme de tri récursif

- L'exemple 18 propose une fonction récursive d'affichage indentée un peu complexe !