



Cours PHP

Licence 2 web (uadb)

Mr Koundoul
birane.koundoul@uadb.edu.sn

Chapitre II:

Les variables

- ✓ **La déclaration des variables**
- ✓ **Les conditions**
- ✓ **Les boucles**
- ✓ **Les dates**
- ✓ **Les tableaux**
- ✓ **Les fonctions**
- ✓ **Inclusion d'une page**

Intégration à HTML

Une page php porte l'extension « .php ». Une page PHP peut être entièrement programmée en PHP ou mélangée avec du code html. PHP est un langage « Embedded HTML », c'est à dire qu'il apparaît à n'importe quel endroit de la page HTML. Pour ça on le place dans des balises particulières : `<?php` et `?>`.

On peut aussi utiliser les balises `<script language="php">` et `</script>`. La première forme est préférable pour plus de simplicité et une compatibilité XHTML. On écrit donc une page HTML dans laquelle on intègre du code PHP.

Exemple:

```
<html>
  <head>
    <title> Premiere Page</title>
  </head>
  <body>
    <?php
      echo "Bonjour les étudiants de web 2";
    ?>
  </body>
</html>
```

Intégration à HTML

Commentaires

Comme en HTML, il est possible de mettre des commentaires dans les pages PHP pour que ce dernier ne soient pas visible au niveau du navigation mais à des utilisation finale. On peut avoir des commentaires sur la même ligne ou sur plusieurs lignes

Exemple

```
<?php
    /* commentaire sur
       plusieurs lignes */
    // Commentaire sur cette ligne
?>
```

Déclaration d'une variable

Type de données

PHP supporte les types de données suivants :

- ✓ nombres entiers,
- ✓ nombres à virgule flottante,
- ✓ chaînes de caractères,
- ✓ tableaux,
- ✓ objets (développés dans la section 'programmation orientée objet').

NB: Tous les noms de variables sont précédés d'un \$

Déclaration d'une variable

Variable

Une variable commence par un dollar « \$ » suivi d'un nom de variable. Les variables ne sont pas typées au moment de leur création. Attention PHP est sensible à la casse : var et Var ne sont pas les mêmes variables ! Voici les règles à respecter :

- ✓ Une variable peut commencer par une lettre
- ✓ Une variable peut commencer par un souligné (underscore) « _ »
- ✓ **Une variable ne doit pas commencer par un chiffre.**

Exemple:

```
<?php
    $var=10;                // déclaration valide
    $Var=2;                // déclaration valide
    $_toto='Bonjour';      // déclaration valide
    $3petitscochons=5;      // Invalide : commence par un chiffre
    $var="Bonjour les étudiants"; // déclaration valide
?>
```

NB: on peut utiliser « " » ou « ' » pour encadrer l'affichage pour echo ou print.

Déclaration d'une variable

Concaténation

On peut déclarer plusieurs variables. Lors de l'affichage, on peut les concaténer en utilisant le point (.).

Exemple 1:

```
<?php
    $prenom="Demba";
    $nom="Thiam";
    echo $prenom.$nom;
?>
```

Pour rendre l'affichage beaucoup plus joli, on peut intégrer les caractères (" ") entre les deux variables pour les séparer.

Exemple 2:

```
<?php
    $prenom="Demba";
    $nom="Thiam";
    echo $prenom." ".$nom;
?>
```

Déclaration de constante

Les constantes

Une constante est un nom qui représente une valeur simple. Par définition la valeur d'une constante ne change pas durant l'exécution du script (sauf pour les constantes magiques). Le langage PHP utilise des constantes scalaires : booléenne (TRUE, FALSE), entières (1,2, -1, ...), réelles (-1.2, 0.314E1,3.14, ...), chaîne de caractères ("CECI est une chaîne de caractères", 'ceci aussi', 'a').

Déclaration d'une constante

Le nom d'une constante comme pour les noms de variables commence par une lettre ou le caractère souligné suivi d'une combinaison de lettres, chiffres ou soulignés. Par convention les caractères utilisés pour nommer une constante sont en majuscule. Les mots clés *define* et *const* permettent de définir une constante. Le mot clé *const* doit toujours être utilisé dans un contexte global.

Déclaration de constante

Exemple:

```
<?php
    define(« universite","Bambey"); //constante universite ayant pour valeur Bambey
    const centre='Diourbel'; //constante centre ayant pour valeur Diourbel
    $a=25;
    if($a<34)
        const responsable='Kasse' ; //contexte local donc erreur...
?>
```

Accès à la valeur une constante

Le nom d'une constante est généralement utilisé pour accéder à sa valeur. Il est également possible d'utiliser la fonction ***constant*** (à utiliser généralement si le nom de la constante n'est connu que durant l'exécution du script.).

Déclaration de constante

Exemple:

```
<?php
    const centre='Diourbel'; //constante CENTRE ayant pour valeur Diourbel
    define("universite","Bambey"); //constante UNIVERSITE ayant pour valeur Bambey
    define("annee",2017); //constante UNIVERSITE ayant pour valeur Bambey

    $moncentre= "centre";
    echo universite;
    echo constant($moncentre);
    echo centre ;
    echo annee;
?>
```

Déclaration de constante

Constantes magiques

PHP fournit des constantes prédéfinies appelées constantes magiques. Certaines constantes appartiennent à des extensions et ne seront disponibles que si ces extensions sont compilées avec PHP. Contrairement aux constantes classiques la valeur d'une constante magique peut changer durant l'exécution d'un script selon le contexte d'utilisation (néanmoins il n'est pas possible de changer dans un script la valeur d'une constante magique.). Il y a sept constantes magiques dont la valeur peut changer selon le contexte :

Nom de la constante	Valeur
<code>__LINE__</code>	La ligne courante dans le fichier.
<code>__FILE__</code>	Le chemin complet et le nom du fichier courant.
<code>__DIR__</code>	Le dossier du fichier.
<code>__FUNCTION__</code>	Le nom de la fonction
<code>__CLASS__</code>	Le nom de la classe courante.
<code>__METHOD__</code>	Le nom de la méthode courante
<code>__NAMESPACE__</code>	Le nom de l'espace de noms courant

Déclaration de constante

Exemple:

```
<?php
    echo __LINE__."<br/>"; //affiche la ligne ou se trouve cette instruction
    echo __LINE__."<br/>"; //affiche la ligne ou se trouve cette instruction donc +1
    echo __FILE__."<br/>"; //affiche le chemin du script.
?>
```

Fonctions pour les constantes

- defined ('nomConstante'): vérifie si une constante a été définie avec define. Elle renvoie une valeur booléenne.
- get_defined_constant(): retourne la liste des constantes avec leurs valeurs

Les opérateurs

Les opérateurs de contrôle

== (2 x =)	strictement égale
!=	différent
>	plus grand que
<	inférieur à
>=	supérieur à
<=	inférieur à
&&	et
	ou
AND	et
OR	ou
TRUE	1 ou oui
FALSE	0 ou non

Les structures de contrôle

Le structure if

L'instruction if est une des plus importantes instructions de tous les langages, PHP inclus ; Elle permet l'exécution conditionnelle d'une partie de code ; Les fonctionnalités de l'instruction if sont les mêmes en PHP qu'en C;

Syntaxe:

if(expression) commande ou { bloc de commandes }

else commande ou { bloc de commandes }

Il y a aussi le « elseif », combinaison du if et du else. Le elseif en un mot peut aussi s'écrire en deux mots : le résultat est le même. On peut écrire des elseif en chaîne. Le premier dont l'expression est vrai est exécuté.

If(expression) commande ou { bloc de commandes }

elseif(expression) commande ou { bloc de commandes }

elseif(expression) commande ou { bloc de commandes }

...

Les structures de contrôle

Le structure if

Exemple:

```
<?php
    $type= "Homme";
    if ($type == "Femme") {
        echo "Bonjour Madame" ;
    } elseif ($type == "Homme") {
        echo "Bonjour Monsieur" ;
    } else {
        echo "Bonjour, je ne connais pas ce genre !" ;
    }
?>
```

Les structures de contrôle

break et continue

L'instruction « **break** » permet de sortir d'un for, while, foreach ou switch. On peut lui indiquer de combien de structures on souhaite sortir si elles sont emboîtées. L'instruction « **continue** » permet de passer à l'itération suivante.

Attention PHP considère le switch comme une boucle, et dans ce cas, réévalue le switch. On peut indiquer à continue combien de structures emboîtées relancer.

Les structures de contrôle

Le structure switch

Le « switch » est équivalent à une série de if et permet de comparer avec un grand nombre de valeurs.

Exemple:

```
<?php
    $i=0;
    switch ($i) {
    case 0: print "i égale 0";
    break;
    case 1: print "i égale 1";
    break;
    case 2: print "i egale 2";
    break;
    default: print "i est inférieur à 0 ou supérieur à 2";
    }
?>
```

Les structures de contrôle

Le structure switch

Le switch s'arrête à la première expression case vraie puis exécute le code suivant dans l'ordre indiqué, jusqu'à la première instruction break. S'il n'y a pas de break, tout le code jusqu'à la fin du switch est exécuté. Dans l'exemple suivant, si \$i vaut 0, tous les print seront affichés !

Exemple

```
<?php
    $i=0;
    switch ($i) {
    case 0: print "i egale 0";
    case 1: print "i egale 1";
    case 2: print "i egale 2";
    default: print "i est inférieur à 0 ou supérieur à 2";
    }
?>
```

NB: Notez aussi que le default doit intervenir en dernier, sinon il n'a aucun intérêt. Enfin on peut employer une syntaxe alternative avec « : » et « endswitch ».

Les structures de contrôle

Le structure switch

```
<?php  
switch ($i):  
    case 0: print "i égale 0";  
    break;  
    case 1: print "i égale 1";  
    break;  
endswitch  
?>
```

Les opérateurs

Les opérateurs mathématique

+	addition
-	soustraction
/	division
*	multiplication

Les boucles

Les boucles

Les boucles vous permettent de parcourir un ensemble d'informations stockées dans un tableau, un fichier texte, une base de donnée, et de les afficher ou de les traiter.

Les différentes boucles sont:

- ✓ while()
- ✓ for();
- ✓ foreach()

Exemple avec while ()

```
$i= 0;  
while ( $i <= 5 )  
{  
    print 'boucle numero '.$i.'<br/>';  
    $i++;  
}
```

Les boucles

Les boucles

Exemple avec while ()

```
<?php
    for ($i=0;$i<=5;$i++)
    {
        echo 'boucle numero '.$i.'<br>';
    }
?>
```

Les boucles

Les boucles

La boucle « foreach » est peut-être l'une des plus intéressantes pour la manipulation de tableaux ou de résultats de requêtes SQL. Elle permet de lister les tableaux. Elle dispose de deux syntaxes.

`foreach (array_expression as $value) commandes`

`foreach (array_expression as $key => $value) commandes`

La première syntaxe récupère les éléments du tableau un par un, séquentiellement. La valeur de l'élément courant du tableau est placée dans `$value`.

La seconde syntaxe est presque identique, sauf qu'en plus la clé (l'index) de l'élément actuel est placée dans `$key`.

Pratique

Exercice 1 :

Créer une page php pour faire la somme de deux nombres.

Exercice 2 :

Créer une page php pour afficher la tableau de multiplication de 3.

Exercice 3:

Créer une page php pour tester si la personne est majeure ou mineure. Si son âge est supérieur à 18, dans ce cas elle est majeure sinon elle est mineure.

Les dates en php

Nous allons apprendre maintenant à manipuler les dates sous différents formats et comment les afficher.

Code à utiliser avec la fonction date() :

format	description	Exemple
a	"am" ou "pm" minuscules	pm
A	"AM" ou "PM" majuscules	PM
d	jour du mois	01 /20
D	jour de la semaine en 3 lettres	mon
F	nom du mois	Janvier
h	heure (format 12 heures avec 0 en entête)	12
H	heure (format 24 heures avec 0 en entête)	08
g	heure (format 12 heures sans 0 en entête)	4
G	heure (format 24 heures sans 0 en entête)	10
i	minutes	44
j	jours du mois (pas de 0 en entête)	3
m	mois de l'année (0 en entête)	04
M	mois de l'année en 3 lettres	jui
n	mois de l'année; pas de 0 en entête	4
s	secondes	30
y	année à 2 chiffres	02
Y	année en 4 chiffres	2002

Les dates en php

La fonction date()

```
<?php
    $date_du_jour = date("d-m-Y");
    echo 'Aujourd\'hui nous sommes le '.$date_du_jour;
?>
```

Résultat : Aujourd'hui nous sommes le 26-04-2017

La date au format système:

Exemple d'affichage avec la fonction time

```
<?php
$date_du_jour = time();
echo 'la date système du jour est '.$date_du_jour;
?>
```

Résultat : la date du système du jour est 1493217107

Les dates en php

Code de tableau à utiliser avec la fonction getdate()

Clés	description	Exemple
seconds	secondes	30
minutes	minutes	5
hours	heures de la journée de 0 à 23	15
mday	jour du mois de 1 à 31	12
wday	jour de la semaine de 0 à 6	2
mon	mois de l'année	4
year	année en 4 chiffres	2002
yday	jour de l'année de 0 à 365	180
weekday	nom du jour de la semaine (en anglais)	monday
month	mois de l'année (en anglais)	january

```
<?php
$time = time(); // la date au format système
$date = getdate($time); // récupération des informations de getdate
echo $date['mday'].' - '.$date['mon'].' - '.$date['year'].' il est '.$date['hours'].':'. $date['minutes'];
?>
```

Pratique

Exercice 1:

Créer une page php pour afficher la date de la machine sur ce format: JJ/MM//AA.

Exercice 2:

Créer une page php pour afficher le temps de la machine h:mn:s

Les tableaux

Tableau

Il existe 2 types de tableaux, les tableaux nominatifs et les associatifs. Pour créer un tableau, utiliser la fonction `array()`;

Un tableau PHP est une association ordonnée. Une association fait correspondre des valeurs à des clés. Les tableaux sont très souples, ils peuvent avoir de multiples dimensions.

Un tableau est créé avec la fonction **`array()`** qui prend comme arguments des paires « key => value » séparées par des virgules. La clé peut être soit un entier soit du texte.

Attention: 8 est un entier, 08 une chaîne ! Si la clé est absente alors c'est la dernière clé entière plus 1 qui est choisie. Si c'est la première, c'est 0 zéro. On accède aux éléments d'un tableau à l'aide des crochets « [et] ». On place entre ces crochets la clé entière ou la chaîne.

Les tableaux

Tableau nominatif ou numérotés

Ces tableaux sont très simples à imaginer. Regardez par exemple ce tableau, contenu de la variable \$prenoms :

\$prenoms est un **array** : c'est ce qu'on appelle une variable "tableau". Elle n'a pas qu'une valeur mais plusieurs valeurs (vous pouvez en mettre autant que vous voulez).

Dans un array, les valeurs sont rangées dans des "cases" différentes. Ici, nous travaillons sur un array numéroté, c'est-à-dire que chaque case est identifiée par un numéro. Ce numéro est appelé **clé**.

```
<?php
```

```
    $prenoms = array ('Demba', 'Fatou', 'Issa');
```

```
?>
```

Clé	Valeur
0	Demba
1	Fatou
2	Issa
.....	

Les tableaux

Tableau nominatif ou numérotés

Ces tableaux sont très simples à imaginer.

Exemple:

```
<?php
    $var=array(10,15,17,23,9);
    echo $var[0]."<br/>";           // 10
    echo $var[3]."<br/>";           // 23
?>
```

L'utilisation de la fonction **array** n'est pas obligatoire et on peut déclarer un tableau à la volée.

```
<?php
    $tab[1]=10;
    $tab[]=6; // équivaut $tab[2]=6
    $tab['message']='le message de bienvenu';
?>
```

Les tableaux

Tableau nominatif ou numérotés

Il est parfois intéressant d'utiliser une boucle pour parcourir les éléments du tableaux.

Exemple:

```
<?php
    $var=array(10,15,17,23,9);
    echo $var[0]."<br/>"; // 10
    echo $var[3]."<br/>"; // 23
    foreach ($var as $key => $value)
    {
        echo $value;
    }
?>
```


Les tableaux

Tableau associatif

Les tableaux associatifs fonctionnent sur le même principe, sauf qu'au lieu de numéroter les cases, on va les étiqueter en leur donnant à chacune un nom différent.

Par exemple, supposons que je veuille, dans un seul array, enregistrer les coordonnées de quelqu'un (nom, prénom, adresse, ville etc...). Si l'array est numéroté, comment savoir que le n°0 est le nom, le n°1 le prénom, le n°2 l'adresse ?... C'est là que deviennent utiles les tableaux associatifs.

Exemple:

```
<?php
```

```
$information= array ( 'prenom' => 'Demba', 'nom' => 'Thiam', 'adresse' =>
'Linguere', 'ville' => 'Louga');
```

```
?>
```

Les tableaux

Tableau associatif

Vous remarquez qu'on écrit une flèche (\Rightarrow) pour dire "associé à". Par exemple, on dit "ville associé à Louga". Nous avons créé un tableau qui ressemble à la structure suivante :

Clé	Valeur
prenom	Demba
nom	Thiam
adresse	Linguere
ville	Louga

Les tableaux

Tableau associatif

Il est aussi possible de créer le tableau case par case comme ceci :

```
<?php
    $information['prenom'] = 'Demba';
    $information['nom'] = 'Thiam';
    $information['adresse'] = 'Linguere';
    $information['ville'] = 'Louga';
?>
```

NB: Pour afficher un élément, il suffit d'indiquer le nom de cet élément entre crochets, ainsi qu'entre guillemets ou apostrophes puisque l'étiquette du tableau associatif est un texte. Par exemple, pour extraire l'adresse, on devra taper :

```
<?php
    echo $information['adresse'];
?>
```

Les tableaux

Parcourir les éléments du tableau

Lorsqu'un tableau a été créé, on a souvent besoin de le parcourir pour savoir ce qu'il contient. Nous allons voir trois moyens d'explorer un array :

- ✓ La boucle for
- ✓ La boucle foreach
- ✓ La fonction `print_r` (utilisée principalement pour le débogage)

Les tableaux

Parcourir les éléments du tableau

La boucle for

Il est très simple de parcourir un tableau numéroté avec une boucle for. En effet, puisqu'il est numéroté à partir de 0, on peut faire une boucle for qui incrémente un compteur à partir de 0 :

Exemple:

```
<?php
    $prenoms = array ('Demba', 'Alioune', 'Issa', 'Fatou');
    for ($i = 0; $i < 4; $i++)
    {
        echo $prenoms[$i] . '<br />';
    }
?>
```

Les tableaux

Parcourir les éléments du tableau

La boucle foreach

La boucle foreach est la boucle plus adaptée aux tableaux. C'est une sorte de boucle for spécialisée dans les tableaux.

foreach va passer en revue chaque ligne du tableau, et lors de chaque passage, il va mettre la valeur de cette ligne dans une variable temporaire.

Exemple:

```
<?php
    $prenoms = array ('Demba', 'Alioune', 'Issa', 'Fatou');
    foreach($prenoms as $pre)
    {
        echo $pre . '<br />';
    }
?>
```

NB: L'avantage de foreach est qu'il permet aussi de parcourir les tableaux associatifs.

Les tableaux

Parcourir les éléments du tableau

La boucle foreach

```
<?php
$information= array ( 'prenom' => 'Demba', 'nom' => 'Thiam', 'adresse' =>
'Linguere', 'ville' => 'Louga');
```

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

NB: Toutefois, avec cet exemple on ne récupère que la valeur. Or, on peut aussi récupérer la clé de l'élément. On doit dans ce cas écrire foreach comme ceci :

```
<?php
    foreach($information as $cle => $valeur)
?>
```

Les tableaux

Parcourir les éléments du tableau

La boucle foreach

```
<?php
$information= array ( 'prenom' => 'Demba', 'nom' => 'Thiam', 'adresse' =>
'Linguere', 'ville' => 'Louga');

foreach($information as $cleinfos => $lesinfos)
{
    echo '[' . $cleinfos . '] vaut ' . $lesinfos. '<br />';
}
?>
```


Les tableaux

Parcourir les éléments du tableau

La fonction print_r

Parfois, en codant votre site en PHP, vous aurez sous les bras un array et vous voudrez savoir ce qu'il contient, juste pour votre information. Vous pourriez utiliser une boucle for ou, mieux, une boucle foreach. Mais si vous n'avez pas besoin d'une mise en forme spéciale et que vous voulez juste savoir ce que contient l'array, vous pouvez faire appel à la fonction print_r. C'est une sorte de echo spécialisé dans les array.

Cette commande a toutefois un défaut : elle ne renvoie pas de code HTML comme `
` pour les retours à la ligne. Pour bien voir les retours à la ligne, il faut donc utiliser la balise HTML `<pre>` qui nous permet d'avoir un affichage plus correct.

Les tableaux

Parcourir les éléments du tableau

La fonction print_r

```
<?php
$information= array ( 'prenom' => 'Demba', 'nom' => 'Thiam', 'adresse' =>
'Linguere', 'ville' => 'Louga');

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

Les tableaux

Rechercher dans un tableau

- ✓ **array_key_exists** : pour vérifier si une clé existe dans l'array
- ✓ **in_array** : pour vérifier si une valeur existe dans l'array
- ✓ **array_search** : pour récupérer la clé d'une valeur dans l'array

Les tableaux

Rechercher dans un tableau

array_key_exists

La fonction **array_key_exists** qui va parcourir le tableau pour nous et nous dire si le tableau contient cette clé. On doit lui donner d'abord le nom de la clé à rechercher, puis le nom de l'array dans lequel on fait la recherche :

Exemple:

```
<?php
$information= array ( 'prenom' => 'Demba', 'nom' => 'Thiam', 'adresse' => 'Linguere', 'ville' =>
'Louga');

if (array_key_exists('nom', $information))
{
    echo 'La clé "nom" se trouve dans les coordonnées !';
}
if (array_key_exists('quartier', $information))
{
    echo 'La clé "quartier" se trouve dans les coordonnées !';
}
?>
```

Les tableaux

Rechercher dans un tableau

in_array

Le principe est le même que array_key_exists... mais cette fois on recherche dans les valeurs. in_array renvoie true si la valeur se trouve dans l'array, false si elle ne s'y trouve pas.

Exemple:

```
<?php
```

```
$information= array ('Demba','Fatou','Issa','Fallou','Aissatou');  
if(in_array('Issa', $information))  
{  
    echo 'La valeur "Issa" se trouve dans les informations !<br/>';  
}  
if(in_array('Aissatou', $information))  
{  
    echo 'La valeur "Aissatou" se trouve dans les informations !<br/>';  
}
```

```
?>
```

Les tableaux

Rechercher dans un tableau

Array_search

array_search fonctionne comme in_array : il travaille sur les valeurs d'un array. Voici ce que renvoie la fonction :

Si elle a trouvé la valeur, array_search renvoie la clé correspondante (c'est-à-dire le numéro si c'est un array numéroté, ou le nom de la clé si c'est un array associatif). Si elle n'a pas trouvé la valeur, array_search renvoie false.

Exemple:

```
<?php
```

```
$information= array ('Demba','Fatou','Issa','Fallou','Aissatou');  
$position = array_search('Issa', $information);  
echo "'Issa' se trouve en position ' . $position . '<br />';  
$position = array_search('Aissatou', $information);  
echo "'Aissatou' se trouve en position ' . $position;
```

```
?>
```

Les tableaux

Tableau multidimensionnel

On peut aussi créer des tableaux multidimensionnels à l'aide des deux méthodes précédentes.

Exemple:

```
<?php
```

```
$stab=array("un"=>array("Demba",1=>"Issa",2=>'Fallou'),2=>array(1,2,3),array('un','deux','trois'));
```

```
echo $stab['un'][0]."<br/>";    // Demba
```

```
echo $stab['un'][1]."<br/>";    // Issa
```

```
echo $stab[2][1]."<br/>";    // 2
```

```
echo $stab[3][2]."<br/>";    // trois
```

```
$stab2['un']['deux']='test';    // créé un tableau à deux dimensions
```

```
echo $stab2['un']['deux'];    // test
```

```
?>
```

Les tableaux

Quelques fonctions avec les tableaux

On peut connaître le nombre d'éléments d'un tableau grâce aux fonctions :

- ✓ `sizeof()` : retourne le nombre d'éléments d'un tableau, ou
- ✓ `count()` : retourne le nombre d'éléments d'un tableau s'il existe, 1 si la variable n'est pas un tableau et 0 si la variable n'existe pas.

Chaque tableau entretient un pointeur courant qui sert à naviguer en son sein grâce aux fonctions :

- ✓ `reset()` : place le pointeur interne sur le **premier élément** et retourne sa valeur,
- ✓ `current()` : retourne la valeur de l'**élément courant** ,
- ✓ `next()` : place le pointeur interne sur l'**élément suivant** et retourne sa valeur,
- ✓ `prev()` : place le pointeur interne sur l'**élément précédent** et retourne sa valeur,
- ✓ `each()` : retourne la **paire clé/valeur courante** du tableau et avance le pointeur sur l'élément suivant (c'est la seule fonction qui ne retourne pas faux si l'élément vaut 0 ou "").

Les tableaux

Quelques fonctions avec les tableaux

Un tableau peut être trié en utilisant les fonctions suivantes :

- ✓ `asort()/arsort()` : trie le tableau en ordre croissant/décroissant de **valeurs**
- ✓ `ksort()/rsort()` : trie le tableau en ordre croissant/décroissant de **clés** ,
- ✓ `sort()` : trie le tableau en ordre croissant **clés et valeurs** (on perd la correspondance clé/valeur),
- ✓ `uasort()/uksort()/usort()` : trie le tableau de la même façon que leurs quasi-homonymes (u pour *user*) mais avec une fonction de comparaison fournie par l'utilisateur.

Pratique

Exercice 1:

Créer un tableau contenant les informations d'un étudiant (matricule, prénom, nom, téléphone).

1. Utiliser la boucle for et foreach pour afficher les informations de l'étudiant.
2. Utiliser la fonction `print_r` avec la balise `<pre>` pour un affichage préformât.
3. Rechercher si une clé et une valeur se trouvent dans le tableau.

Les fonctions

Une fonction est une série d'instructions qui effectue des actions et qui retourne une valeur. En général, dès que vous avez besoin d'effectuer des opérations un peu longues dont vous aurez à nouveau besoin plus tard, il est conseillé de vérifier s'il n'existe pas déjà une fonction qui fait cela pour vous. Et si la fonction n'existe pas, vous avez la possibilité de la créer.

Comme en langage C, il existe déjà des fonctions en php permettant de résoudre des tâches spécifiques.

Les fonctions

PHP propose des centaines et des centaines de fonctions prêtes à l'emploi. Voici un petit aperçu des fonctions qui existent en php :

- ✓ Une fonction qui permet de rechercher et de remplacer des mots dans une variable
- ✓ Une fonction qui envoie un fichier sur un serveur
- ✓ Une fonction qui envoie un mail avec PHP (très pratique pour faire une newsletter !)
- ✓ Une fonction qui crypte des mots de passe.
- ✓ Une fonction qui renvoie l'heure, la date...
- ✓ etc.

Les fonctions

Voici une liste de fonction:

Fonction	Description
strtolower()	Mise en minuscule
strtoupper()	Mise en majuscule
ucfirst()	Mise en majuscule de l'initiale
nl2br()	Remplace le \n par pour affichage
htmlspecialchars()	Convertit les caractères html
addslashes()	Fait précéder les caractères spéciaux d'un \
stripslashes()	Supprime les \
ltrim()	Supprime les espaces initiaux
trim()	Supprime les espaces en début et fin

Les fonctions

PHP dispose de nombreuses fonctions mathématiques qui manipulent et produisent des nombres entiers et décimaux. Le tableau suivant en liste quelques unes avec des exemples d'utilisation.

Fonctions	Exemple	Résultat	Description
abs	abs(-5)	5	valeur absolue d'un nombre
ceil	ceil(2.4)	3	entier supérieur
floor	floor(2.6)	2	entier inférieur
fmod	fmod(7.2,2)	1.2	reste decimal d'une division à résultat entier
max, min	max(10,12.3,1)	12.3	plus grande valeur (utilisation possible d'un tableau).
pow	pow(10,2)	100	puissance d'un nombre
rand	rand(1,9)	3	valeur aléatoire entre un min (0) et un max optionnels
round	round(1.3481,2)	1.35	valeur arrondie avec une précision optionnelle
sqrt	sqrt(9)	3	racine carrée

Les fonctions

Exemple :

```
<?php
    $prenom="Demba";
    echo strtolower($prenom)."<br/>"; // pour mettre en minuscule
    echo strtoupper($prenom)."<br/>"; // pour mettre en majuscule
    echo ucfirst($prenom)."<br/>"; // pour mettre le premier caractère en majuscule
?>
```

NB: les fonctions ltrim() et trim() sont fréquemment utilisées avec les formulaires pour éviter les espaces lors de l'enregistrement dans une base de données ou fichiers.

Les fonctions

strlen

Cette fonction retourne la longueur d'une chaîne de caractères, c'est-à-dire le nombre de lettres et chiffres qu'il y a (espaces compris).

Exemple :

```
<?php
    $phrase = 'je veux compter le nombre de caracteres';
    $longueur = strlen($phrase);
    echo 'La phrase compte ' . $longueur . ' caractères';
?>
```

Résultat:

La phrase compte 39 caractères

Les fonctions

str_replace

str_replace remplace une chaîne de caractères par une autre.

Exemple :

```
<?php
    $remplace = str_replace('b', 'd', 'bemba boubou');
    echo $remplace;
?>
```

Résultat :

demda doudou

NB: la liste des fonctions est exhaustive.

Les fonctions

str_shuffle

Cette permet de mélanger les caractères d'un mot ou d'une phrase donné.

Exemple:

```
<?php
    $chaine = 'Cette chaine va etre melangee !';
    $chaine = str_shuffle($chaine);
    echo $chaine;
?>
```

Les fonctions

Bien que PHP propose des centaines et des centaines de fonctions. Parfois il n'y aura pas ce que vous cherchez et il faudra écrire vous-même la fonction. C'est une façon pratique d'étendre les possibilités offertes par PHP.

En général, si vous effectuez des opérations un peu complexes que vous pensez avoir besoin de refaire régulièrement, il est conseillé de créer une fonction.

Pour créer une fonction, il vous faut le mot clé **function** suivi du nom de la fonction. Elle peut ne pas prendre de paramètres comme elle peut en avoir un à plusieurs.

Exemple sans paramètre:

```
<?php
function ma_fonction()
{
    traitement;
}
?>
```




```
<?php
function somme()
{
    $a=5;
    $b=6;
    $s=$a+$b;
    return "La somme est " . $s;
}
echo somme();
?>
```

Les fonctions

Exemple avec paramètre:

```
<?php
function ma_fonction($var1,$var2,...,$varn)
{
    traitement;
}
?>
```



```
<?php
function somme($a,$b)
{
    $s=$a+$b;
    return "La somme est " . $s;
}
echo somme(8,9);
?>
```

NB: on peut avoir une fonction sans retour.

Pratique

Exercice 1:

Créer une page php permettant de remplacer tous les caractères e en a dans une phrase donnée.

Exercice 2:

Utiliser les fonctions php pour mettre :

- ✓ Le premier caractère de la phrase en majuscule
- ✓ Un mot de la phrase en majuscule
- ✓ Les premiers caractères de chaque mot en majuscule

Exercice 3:

Ecrire votre propre fonction pour calculer la surface du cercle en précisant les paramètres dans la fonction

Inclusion d'une page

« **require** » et « **include** » incluent à l'endroit actuel et exécutent le fichier PHP. Ils sont identiques dans leur fonctionnement à une exception : le traitement des erreurs. Un **include** produit un « **warning** » (le code continue en principe à s'exécuter) tandis qu'un **require** produit une « **erreur fatale** » (l'exécution s'arrête).

Comme **require** et **include** sont des éléments du langage et pas des fonctions il y a pas besoin d'utiliser les parenthèses. « **require_once** » et « **include_once** » ressemblent à leurs homologues avec cependant une différence. Quoi qu'il arrive, le fichier est inclus une seule fois. Si un second « **require_once** » apparaît avec le même fichier, l'instruction ne sera pas exécutée.

Inclusion d'une page

Include

- ✓ Semblable aux include du C/C++
- ✓ Réalise une inclusion physique du fichier demandé

include_once :

- ✓ identique au include
- ✓ protège contre d'éventuelles inclusions multiples
- ✓ qui pourraient mener à des erreurs (redéclarations, etc.)

require et require_once :

- ✓ fonctionnent comme le include et le include_once respectivement
- ✓ mais le programme s'arrête si le fichier inclus n'existe pas

Inclusion d'une page

```
<?php
    include("indexs.php");
    echo "<br/>";
    echo "je suis un autre
    texte";
?>
```



```
<?php
    require("indexs.php");
    echo "<br/>";
    echo "je suis un autre
    texte";
?>
```