

# COURS DE PROGRAMMATION WEB DYNAMIQUE



# PARTIE 1 :

## Bases de la programmation en PHP

# PARTIE 1 :

## Les principes de base

# PRINCIPES DE BASE

- PHP utilise les notions de base de la programmation : variables, séquentialité, conditions, boucles, tableaux, fonctions ...
- Toute instruction (sauf les structures de contrôle et sauf la dernière ligne de code) doit se terminer par un " ; "
- Un script php se commente comme en C :
  - **//** : pour une seule ligne
  - **/\* \*/** : multilignes
  - **#** : pour une seule ligne

# PRINCIPES DE BASE

## Sensibilité à la casse. En règle générale :

- Les noms de variables sont sensibles à la casse
- Les autres éléments du langage (fonctions, instructions...) ne le sont pas.
- *exemple* : print , Print , pRint , prinT ... toutes acceptées
- *Important* : il est fortement recommandé de respecter la casse dans :
  - tous les liens HTML
  - les noms des tables MySQL (ce sont des fichiers)

Les blocs d'instructions (une à plusieurs instructions) peuvent être identifiés, délimités par des accolades " { } "

# LES INSTRUCTIONS D’AFFICHAGE

- les fonctions: echo - print()

la commande echo() est une construction de php.  
Elle permet d'afficher à l'écran des chaînes de caractères, qui peuvent être définies directement par l'utilisateur ou qui peuvent être des contenus de variables.

*exemple :*

```
$nom="said" ;
```

```
echo $nom ;  
echo 'Salam' ;  
echo "Salam" ;
```

```
print $nom ;  
print 'Salam' ;  
print "Salam" ;
```

echo 'Salam',\$nom ; plusieurs paramètres séparées par " , "

# LES INSTRUCTIONS D’AFFICHAGE

- les fonctions: echo - print()  
On peut aussi les écrire avec des ( )

*exemple :*

```
<?php
    $nbr=1;
    echo($nbr);
    echo('bonjour');
    echo("bonjour");
?>
```

```
<?php
    $nbr=1;
    print($nbr);
    print('bonjour');
    print("bonjour");
?>
```

# LES INSTRUCTIONS D’AFFICHAGE

- les fonctions: echo - print()

Utiliser " " ou ' '

- ' ' et " " sont utilisés pour délimiter des chaînes de caractères.
- PHP examinera ce que contient une chaîne entre " ", mais pas une chaîne qui est entre ' ' qu'il affichera directement.  
==> ' ' annule le remplacement des variables par leurs valeurs.
- **echo "Salam \$nom";** ==> affiche **Salam Said**  
*la variable est remplacée par sa valeur.*
- **echo 'Salam \$nom' ;** ==> affiche **Salam \$nom**
- **print 'Salam \$var' ;** ==> affiche **Salam \$var**

*\$nom et \$var sont interprétées comme des chaînes de caractères.*



# VARIABLES, TYPES ET OPÉRATEURS

- Il existe différents types de variables en PHP:
  - ✓ les variables définies par le programmeur ;
  - ✓ les variables d'environnement ;
  - ✓ les variables de sessions etc ...
- Les variables commencent obligatoirement par **\$** puis une **lettre** ou un **\_**, puis une suite de lettres, de chiffres ...
- *exemple :*  
\$1var : écriture fausse                      \$\_1var : écriture juste
- Par convention, un nom de variable ne commence pas par une majuscule. S'il faut plusieurs mots pour composer le nom, ils sont habituellement séparés par des soulignés (\_).

# VARIABLES, TYPES ET OPÉRATEURS

## Déclaration de variables:

- PHP ne contient pas de partie déclarative clairement définie. Pour déclarer une variable, il suffit de l'initialiser. Dès lors elle est accessible.
- en PHP la déclaration de variables est implicite, c'est-à-dire la déclaration se fait en attribuant une valeur à la variable.
- en PHP, les variables ne sont pas typées.  
C'est le contenu de la variable, en cours de traitement, qui permettra d'assigner à la variable le type le plus approprié.
- On parle de chargement de variables, et on appelle cette technique le "typage dynamique".

# VARIABLES, TYPES ET OPÉRATEURS

exemple :  
<?php

```
$var=1 ;
```

```
print ($var);
```

==> affiche 1 (var de type int)

```
$var="salam" ;
```

```
print ($var);
```

==> affiche salam (ch. de caract.)

```
$var=1.66 ;
```

```
print $var;
```

==> affiche 1.66 (type double)

```
?>
```

# VARIABLES, TYPES ET OPÉRATEURS

## Les types des variable en PHP

- **boolean** : "vrai" ou bien "faux", "1" ou "0" ...
- integer** : valeur numérique entière
- float-double** : valeur numérique flottante (à virgule)
- string** : chaîne de caractères (texte)
- array** : tableau (ensemble de valeurs)
- object** : objet (instance de classe) utilisé en POO
- resource** : une ressource (type abstrait, inutilisable par le programmeur, utilisé uniquement pour des fonctions). Par exemple une variable permettant d'identifier une connexion à une base de données.
- NULL** : désigne l'absence de valeur : valeur/variable vide ou inexistante

# VARIABLES, TYPES ET OPÉRATEURS

- On peut incruster une variable dans une autre variable:

```
$var1 = 'wa sahlane!';
```

```
$var2 = "Ahlane $var1";
```

```
echo $var2; // Affiche: Ahlane wa sahlane!
```

- Une variable peut avoir pour identificateur la valeur d'une autre variable.

**Syntaxe :**

```
${$var} = valeur;
```

**exemple :**

```
$Mois = "Avril";
```

```
${$Mois} = 4;
```

```
echo $Avril; // la variable $Avril vaut 4
```

# VARIABLES, TYPES ET OPÉRATEURS

## Les chaînes de caractères:

- Une variable chaîne de caractères n'est pas limitée en nombre de caractères. Elle est toujours délimitée par des simples quotes ou des doubles quotes.

exemple :

```
$nom = "Botoel";  
$prenom = 'walid';
```

- Comme on a vu les doubles quotes permettent l'évaluation des variables et caractères spéciaux contenus dans la chaîne (comme en C) alors que les simples ne le permettent pas.

# VARIABLES, TYPES ET OPÉRATEURS

## Les Opérateurs :

Les opérateurs sont les mêmes que ceux utilisés dans C :

- Les opérateurs arithmétiques : **+** , **-** , **\*** , **/** , **%**
- Les Opérateurs unaires : **++** , **--**
- Les opérateurs logiques : **&&** , **||** , **!**
- Les opérateurs de comparaison : **==** , **!=** , **>** , **<** , **>=** , **<=**
- L'Opérateur ternaire/conditionnel :  
**(condition) ? (expr1) : (expr2)**
- Les opérateurs d'affectation ...

# VARIABLES, TYPES ET OPÉRATEURS

- Les Opérateurs :

**exemple:**

```
<?php
```

```
    $a=4 ; $b=1 ; $c=-2 ; $d=2 ; $x=3 ;
```

```
    $a += ($x+5) ;
```

```
    echo($a!=$c*=(-$d));
```

```
    $a*=$c+($x-$d) ;
```

```
    $a %= $d++ ;
```

```
    $a %= ++$d ;
```

```
    echo '$a='.$a.' $c='.$c.' $x='.$x.' $d='.$d.'<br>';
```

```
?>
```



# VARIABLES, TYPES ET OPÉRATEURS

- L'Opérateur de concaténation de chaînes : . (point)

Il est utilisé pour concaténer des chaînes, variables etc ...

**exemple :**

- print("Il est \$date" . "gmt");

*// \$date est une variable et gmt un texte*

- <?php

\$nom = " Said "; \$i=1;

echo 'Mon nom est ' . \$nom.'<br>';

*// affiche : Mon nom est Said puis saut de ligne*

print '<br>test numero: ' . ++\$i.'<br>';

*// incrémentation, saut de ligne puis affichage*

?>

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions utiles sur les variables :

**empty(\$var)** : renvoie vrai si la variable est vide

**isset(\$var)** : renvoie vrai si la variable existe

**unset(\$var)** : détruit une variable

**gettype(\$var)** : retourne le type de la variable

**settype(\$var, "type")** : permet de modifier le type d'une variable pendant l'exécution du programme au type "**type**" (cast)

**var\_dump()** : affiche le type d'une variable et son contenu (ainsi que sa taille si c'est une chaîne)

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions utiles sur les variables :

`is_numeric()`  
`is_int()`  
`is_integer()`  
`is_long()`  
`is_double()`  
`is_real()`  
`is_float()`  
`is_string()`  
`is_array()`  
`is_object()`  
`is_bool()`  
`is_null()`  
`is_resource()`  
`is_scalar()`

La variable est de type numérique ?

La variable est un entier ?

La variable est un nombre décimal ?

La variable est un chaîne de caractères ?

La variable est une liste ?

La variable est un objet ?

La variable est un booléen ?

La variable est nulle ?

Indique si la variable est une ressource.

La variable est scalaire ?

La réponse à toutes ces questions sera "Vrai" ou "Faux".

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions utiles sur les variables :

*exemples:*

<?php

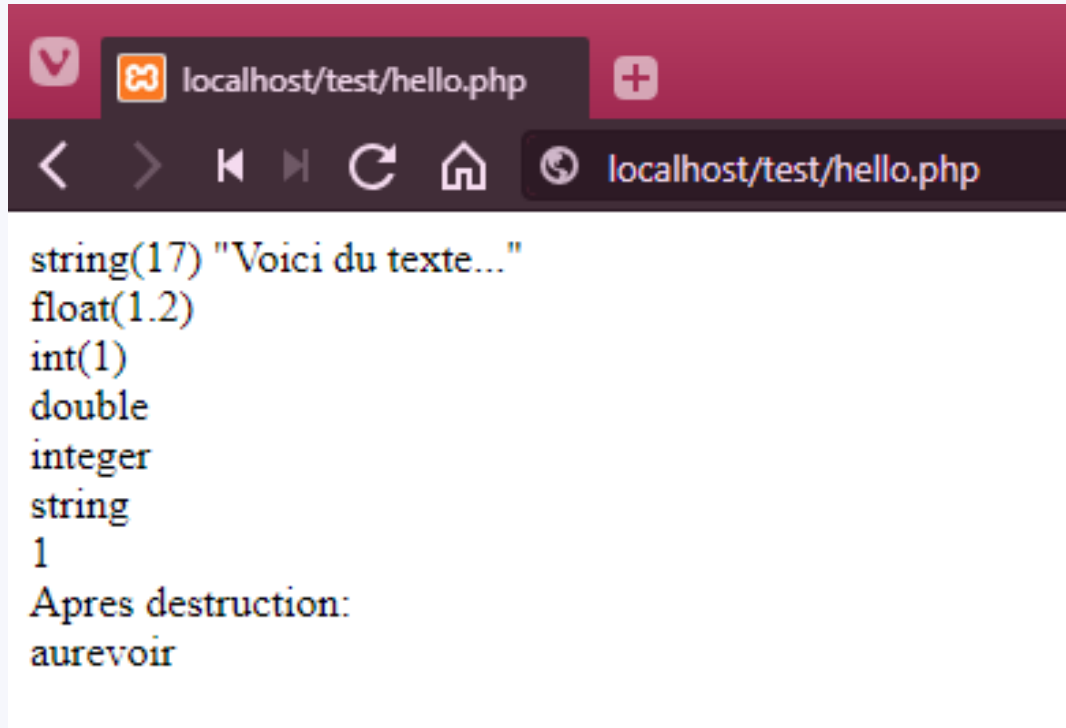
```
echo var_dump("Voici du texte...").'<br>';  
echo(var_dump(1.2).'<br>');  
print(var_dump(1)).'<br>';  
$var=8.9;  
echo gettype($var).'<br>';  
$var=9;  
echo gettype($var).'<br>';  
$var='texte';  
echo gettype($var).'<br>';  
echo isset($var);  
unset ($var);  
echo ("<br> Apres destruction: ".isset($var).'<br>');  
echo "aurevoir";
```

?>

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions utiles sur les variables :

*exemples: affichage*



```
string(17) "Voici du texte..."
float(1.2)
int(1)
double
integer
string
1
Après destruction:
aurevoir
```

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions sur les chaînes :

On dispose de plusieurs fonctions prédéfinies pour effectuer diverses opérations sur les chaînes de caractères.

- Recherche.
- Comparaison.
- Extraction.
- Remplacement.
- Suppression.
- Ajout.

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions sur les chaînes :

**\$resultat= chop(\$chaine);**

Supprime les espaces blancs en fin de chaîne.

**\$liste = explode(\$delimiteur, \$chaine);**

Transforme une chaîne de caractères en une liste.

**\$chaine\_result = implode(\$delimiteur, \$liste);**

Concatène tous les éléments d'une liste dans une chaîne.

**\$nombre = strpos(\$chaine, \$caractere [, \$depart]);**

Retourne la position de la première occurrence d'un caractère dans une chaîne.

**\$nombre = strrpos(\$chaine, \$caractere);**

Position de la dernière occurrence d'un caractère dans une chaîne.

**\$chaine\_result = strrev(\$chaine);**

Inversion de l'ordre des caractères d'une chaîne

# VARIABLES, TYPES ET OPÉRATEURS

- Fonctions sur les chaînes :

**\$chaine\_result = strtolower(\$chaine);**

Transforme tous les caractères d'une chaîne en minuscules.

**\$chaine\_result = strtoupper(\$chaine);**

Transforme tous les caractères d'une chaîne en majuscules.

**\$chaine\_result = substr(\$chaine, \$position, \$longueur);**

Retourne une sous chaîne de \$chaine.

**\$nombre = substr\_count(\$chaine, \$sous-chaîne);**

Retourne le nombre de sous chaînes trouvées dans la chaîne de caractères.

**\$chaine\_result = ucfirst(\$chaine);**

Retourne la chaîne de caractères avec son premier caractère en majuscule.

**\$chaine\_result = ucwords(\$chaine);**

Retourne la chaîne de caractères avec chaque premier caractère de mot en majuscule.



# VARIABLES, TYPES ET OPÉRATEURS

- **exercice :**

**Ecrire un programme qui déclare trois variables \$prix\_livre, \$prix\_revue et \$nombre, les initialise respectivement à 150, 100 et 10, calcule le prix hors taxe des 10 revues, et compare le prix d'une revue et d'un livre et affiche quel est le prix le plus élevé.**

# VARIABLES, TYPES ET OPÉRATEURS

- exercice :

```
<?php
    $prix_livre = "150" ;
    $prix_revue = "100" ;
    $Nombre = "10" ;
    echo '- $prix_livre='.$prix_livre." <br/>" ;
    echo "- $prix_livre= $prix_livre <br/>" ;
    $livre_HT=$prix_livre*$Nombre;
    echo 'Prix Total='.$livre_HT.'<br>';
    $revue_HT=$revue_livre*$Nombre;
    print 'Le prix le plus élevé est celui de: ';
    if($prix_livre>$prix_revue)
        print 'le livre';
    else
        print 'la revue';
?>
```

# VARIABLES, TYPES ET OPÉRATEURS

- Variables spéciales php:

- **\$\_POST**: Les valeurs envoyées par formulaire ;
- **\$\_GET**: Les valeurs provenant de l'URL ;
- **\$\_FILES**: Les fichiers envoyés par formulaire ;
- **\$\_ENV**: contient les noms et les valeurs des variables d'environnement (syst. exploitation). C'est le plus souvent sous des serveurs Linux que l'on retrouve des informations dans cette superglobale.
- **\$GLOBALS**: contient les valeurs de toutes les variables globales du script.
- **\$\_SERVER**: contient les informations liées au serveur Web.
- **\$\_COOKIE**: contient le nom et la valeur des cookies enregistrés sur le poste client.
- **\$\_SESSION**: contient l'ensemble des noms des variables de session et leurs valeurs

# CONSTANTES

La déclaration des constantes se fait au moyen de la fonction `define()`

***Syntaxe:***

```
define("var",val)
```

définit la constante **var** de valeur **val**

- Les constantes ne portent pas le symbole \$ (dollars) en début d'identificateur et ne sont pas modifiables.
- A la déclaration, si la valeur d'une constante est une chaîne de caractères, il faut l'entourer d'apostrophes ou de guillemets doubles.

# CONSTANTES

## exemple 1 :

```
define("Nom","Mansouri");  
echo Nom;                // affiche Mansouri
```

## exemple 2 :

```
define('Naissance',1990);  
echo Naissance;          // affiche 1990
```

# CONSTANTES

- Contrairement aux variables, les identificateurs de constantes (comme ceux des fonctions) ne sont pas sensibles à la casse.
- Les constantes sont définies et accessibles à tout endroit du code, globalement.

# CONSTANTES

- exemple :

Utilisation des constantes dans des expressions

**<?php**

```
define("pi",3.14159265);
define('e',2.71828183);
define('unite',' metres');
$diametre = 10 ;
$perimetre = $diametre * pi;
print ("Perimetre = ".$perimetre.unite."\n");
$surface = ($diametre * $diametre) / 4 * pi;
print ("Surface = ".$surface.unite." carres.\n");
```

**?>**

# LES VARIABLES D'ENVIRONNEMENT

- PHP propose toute une série de variables qui sont déjà implantées dans le langage sans qu'on ait à les créer. On les appelle les variables d'environnement.
- Ces variables sont des données stockées dans des variables permettant au programme d'avoir des informations sur son environnement (serveur et client).
- Elles permettent notamment d'avoir des informations sur le type de serveur, son administrateur, la date à laquelle le script a été appelé, l'adresse IP et le type de navigateur du client ....



# LES VARIABLES D'ENVIRONNEMENT

- Ces variables sont créées par le serveur à chaque fois que le script PHP est appelé, le serveur les lui fournit en paramètres cachés lors de l'exécution de l'interpréteur.
- Ces variables appartiennent à la famille des variables globales `$_SERVER`
- La liste de ces variables est obtenue à partir de la fonction `phpinfo()`. (accès à partir du code ou icône ...)

```
<?php  
    phpinfo();  
?>
```

**dans la page affichée les variables se trouvent sous le titre: php Variables**

# LES VARIABLES D'ENVIRONNEMENT

PHP Variables

| Variable                                    | Value   |
|---|---|
| \$_SERVER['MIBDIRS']                        | C:/xampp/php/extras/mibs  |
| \$_SERVER['MYSQL_HOME']                     | \xampp\mysql\bin  |
| \$_SERVER['OPENSSL_CONF']                   | C:/xampp/apache/bin/openssl.cnf   |
| \$_SERVER['PHP_PEAR_SYSCONF_DIR']           | \xampp\php  |
| \$_SERVER['PHPRC']                          | \xampp\php  |
| \$_SERVER['TMP']                            | \xampp\tmp  |
| \$_SERVER['HTTP_HOST']                      | localhost   |
| \$_SERVER['HTTP_CONNECTION']                | keep-alive  |
| \$_SERVER['HTTP_CACHE_CONTROL']             | max-age=0   |
| \$_SERVER['HTTP_UPGRADE_INSECURE_REQUESTS'] | 1   |
| \$_SERVER['HTTP_USER_AGENT']                | Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.103 Safari/537.36 Vivaldi/2.1.1337.47  |
| \$_SERVER['HTTP_ACCEPT']                    | text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8   |
| \$_SERVER['HTTP_REFERER']                   | http://localhost/test/  |
| \$_SERVER['HTTP_ACCEPT_ENCODING']           | gzip, deflate   |
| \$_SERVER['HTTP_ACCEPT_LANGUAGE']           | en-US,en;q=0.9  |
| \$_SERVER['PATH']                           | C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Users\whitedragon\AppData\Local\Programs\Microsoft VS Code\bin |
| \$_SERVER['SystemRoot']                     | C:\Windows  |
| \$_SERVER['COMSPEC']                        | C:\Windows\system32\cmd.exe   |
| \$_SERVER['PATHEXT']                        | .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC   |
| \$_SERVER['WINDIR']                         | C:\Windows  |
| \$_SERVER['SERVER_SIGNATURE']               | <address>Apache/2.4.34 (Win32) OpenSSL/1.1.0i PHP/7.2.9 Server at localhost Port 80</address>   |
| \$_SERVER['SERVER_SOFTWARE']                | Apache/2.4.34 (Win32) OpenSSL/1.1.0i PHP/7.2.9  |
| \$_SERVER['SERVER_NAME']                    | localhost   |

# LES VARIABLES D'ENVIRONNEMENT

- exemples :

**`$_SERVER["HTTP_HOST"] :`**

affiche l'hôte, c' est à dire le nom de votre espace web

**`$_SERVER["HTTP_REFERER"] :`**

affiche la provenance de votre visiteur, l'url d' ou celui ci arrive

**`$_SERVER["DOCUMENT_ROOT"] :`**

le répertoire racine de l'arborescence des document sur le serveur

**`$_SERVER["REQUEST_METHOD"] :`**

la méthode utilisée, GET, POST, pratique pour vérifier les variables provenant d'un formulaire.

**`$_SERVER["REMOTE_ADMIN"] :`**

donne l'adresse de l'administrateur du serveur.

**`$_SERVER["REMOTE_ADDR"] :`**

donne l'adresse IP du client ou visiteur.