



Syntaxe de base du Langage PHP

Enseignante:

Mme Imène Jemmali & Mme Dalila Amara

Plan

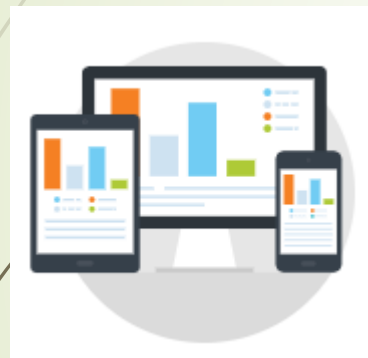
- Rappel
- Les bases de PHP
- Les constantes
- Les variables
- Les types de données
- La conversion des types
- La concaténation
- Les opérateurs
- Les instructions
- Les fonctions
- Les tableaux

Rappel

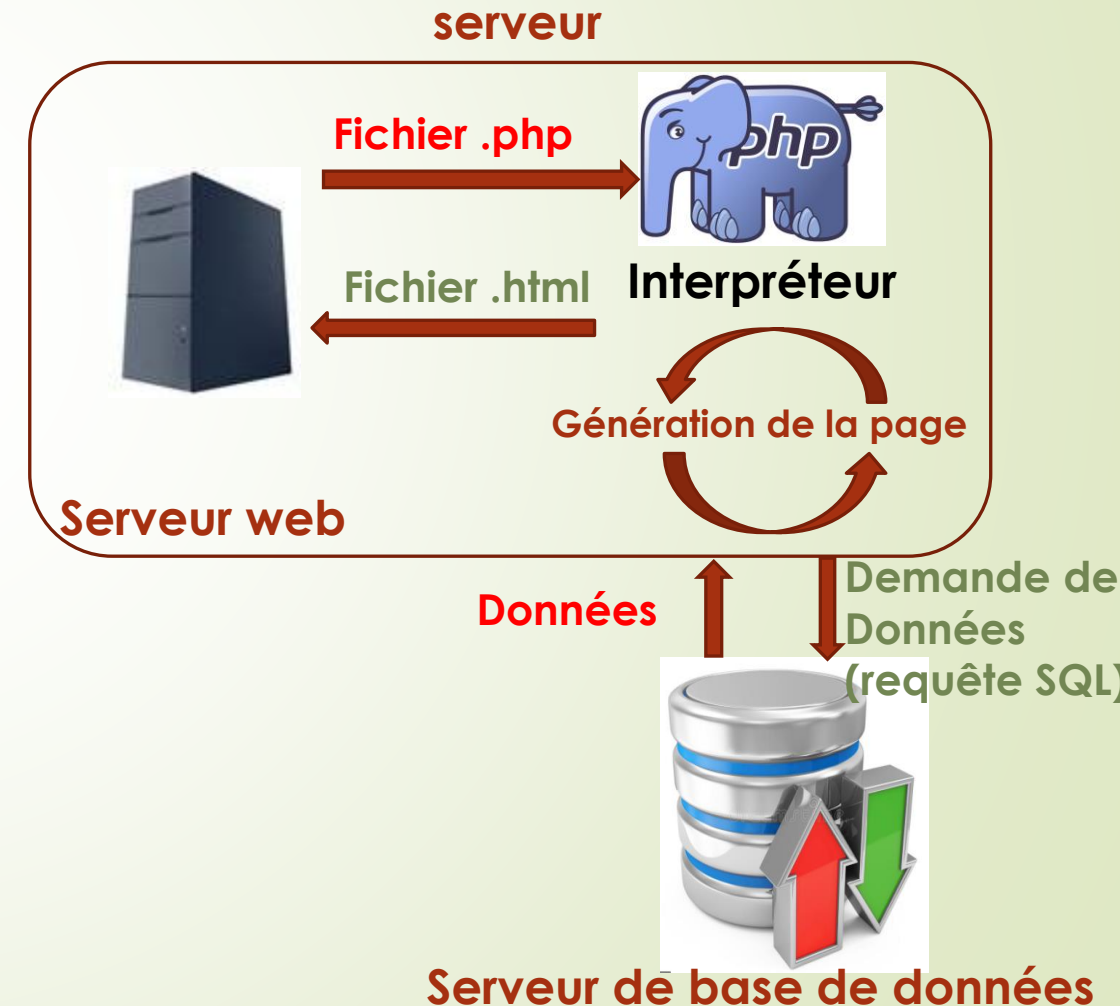
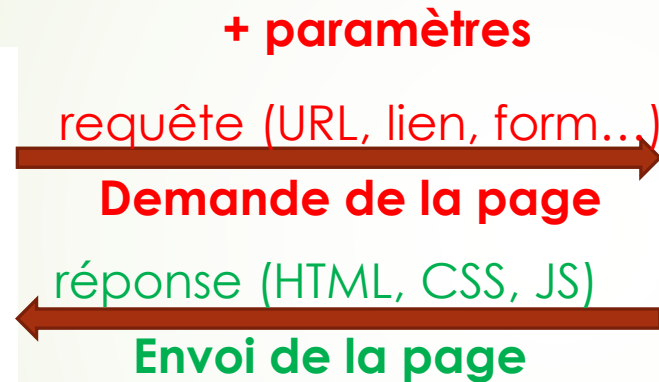


WampServer

- Serveur web: Apache
- PHP
- Base de donnée: MySQL



Client



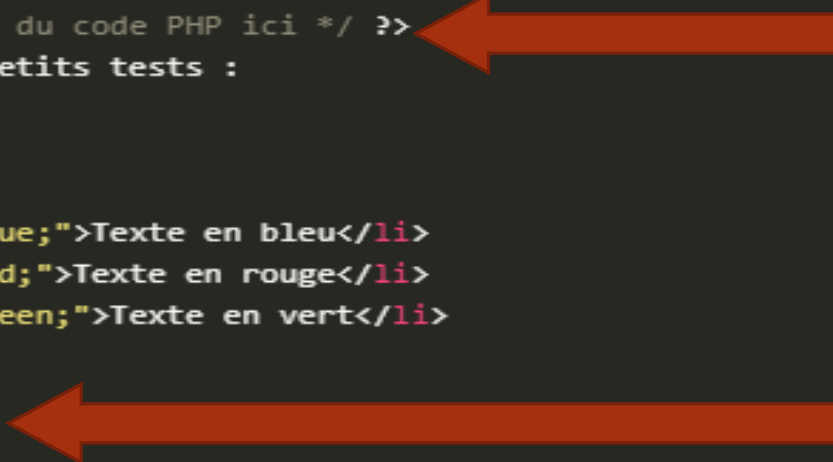
Les balises PHP

→ `<? ?>`
→ `<% %>`

- Les pages contenant du code PHP ont l'extension **.php**
- Une page PHP est **généralement** une simple page HTML qui contient des instructions en langage PHP
- Ces bouts de code PHP seront les parties dynamiques de la page
- La balise PHP est de la forme suivante :
`<?php` /* Le code PHP se met ici */ **`?>`**
- On peut placer une balise PHP n'importe où dans le code (dans l'en-tête de la page , au milieu d'une balise HTML ...)
- Toute instruction se termine par un « ; »
- Les commentaires:
 - **`//commentaire monoligne`**
 - **`/* commentaire multiligne */`**

Exemple:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Ceci est une page de test avec des balises PHP</title>
5     <meta charset="utf-8" />
6   </head>
7   <body>
8     <h2>Page de test</h2>
9
10    <p>
11      Cette page contient du code HTML avec des balises PHP.<br />
12      <?php /* Insérer du code PHP ici */ ?>
13      Voici quelques petits tests :
14    </p>
15
16    <ul>
17      <li style="color: blue;">Texte en bleu</li>
18      <li style="color: red;">Texte en rouge</li>
19      <li style="color: green;">Texte en vert</li>
20    </ul>
21
22    <?php
23      /* Encore du PHP
24      Toujours du PHP */
25    ?>
26  </body>
27 </html>
```



Les constantes

Exemple1: Définir une constante en utilisant la fonction **define**

```
<?php
define("CONSTANTE", "Bonjour le monde.");
echo CONSTANTE; // affiche "Bonjour le monde."
echo Constante; // affiche "Constante" et une notice.
?>
```

Remarque: Les constantes prédéfinies

- NULL, _FILE_, ...
- PHP_VERSION
- PHP_OS
- TRUE et FALSE
- E_ERROR

Exemple2: Définir une constante en utilisant le mot clé **const**

```
<?php
// Fonctionne depuis PHP 5.3.0
const CONSTANT = 'Bonjour le monde !';

echo CONSTANT;

// Fonctionne depuis PHP 5.6.0
const ANOTHER_CONST = CONSTANT.'; Aurevoir le monde !';
echo ANOTHER_CONST;

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

// Fonctionne depuis PHP 7
define('ANIMALS', array(
    'chien',
    'chat',
    'oiseaux'
));
echo ANIMALS[1]; // affiche "chat"
?>
```

Les variables

- Petite information stockée en mémoire temporairement
- Représenter par un signe dollar "\$" suivi du nom de la variable
- Commencer par **une lettre** ou **un souligné (_)**, suivi de lettres, chiffres ou soulignés
- Pas besoin d'être déclarée (**typage implicite**)

Exemple:

```
<?php
$var = 'Jean';
$Var = 'Paul';
echo "$var, $Var";           // affiche "Jean, Paul"

$4site = 'pas encore';      // invalide : commence par un nombre
$_4site = 'pas encore';     // valide : commence par un souligné
?>
```


Les variables

Les fonctions de gestion des variables:

`empty(), gettype(), intval(), is_array(), is_bool(),
is_double(), is_float(), is_int(), is_integer, is_long(),
is_object(), is_real(), is_numeric(), is_string(), isset(),
settype(), unset() ...`

Les variables

Visibilité des variables:

- Variable **locale**

Visible uniquement à l'intérieur d'un contexte d'utilisation

- Variable **globale**

Visible dans tout le script

Utilisation de l'instruction **global** dans des contextes locaux

- Variable **superglobale**

Les variables

➤ Variables prédéfinies

➤ Les variables d'environnement dépendant du client

<i>Variable</i>	<i>Description</i>
<code>\$_SERVER["HTTP_HOST"]</code>	Nom d'hôte de la machine du client (associée à l'adresse IP)
<code>\$_SERVER["HTTP_REFERER"]</code>	URL de la page qui a appelé le script PHP
<code>\$_SERVER["HTTP_ACCEPT_LANGUAGE"]</code>	Langue utilisée par le serveur (par défaut en-us)
<code>\$_SERVER["HTTP_ACCEPT"]</code>	Types MIME reconnus par le serveur (séparés par des virgules)
<code>\$_SERVER["CONTENT_TYPE"]</code>	Type de données contenu présent dans le corps de la requête. Il s'agit du type MIME des données
<code>\$_SERVER["REMOTE_ADDR"]</code>	L'adresse IP du client appelant le script CGI
<code>\$_SERVER["PHP_SELF"]</code>	Nom du script PHP

Les variables

▀ Variables prédéfinies

▀ Les variables d'environnement dépendant du serveur

Variable	Description
<code>\$_SERVER["SERVER_NAME"]</code>	Le nom du serveur
<code>\$_SERVER["HTTP_HOST"]</code>	Nom de domaine du serveur
<code>\$_SERVER["SERVER_ADDR"]</code>	Adresse IP du serveur
<code>\$_SERVER["SERVER_PROTOCOL"]</code>	Nom et version du protocole utilisé pour envoyer la requête au script PHP
<code>\$_SERVER["DATE_GMT"]</code>	Date actuelle au format GMT
<code>\$_SERVER["DATE_LOCAL"]</code>	Date actuelle au format local
<code>\$_SERVER["\$DOCUMENT_ROOT"]</code>	Racine des documents Web sur le serveur

Les types de données (1/3)

- Pas besoin d'affecter un type à une variable avant de l'utiliser
- La même variable peut **changer de type** en cours de script
- Les variables issues de l'envoi des données d'un formulaire sont du type **string**

Les types de données (2/3)

- Les différents types de données
 - Les entiers : le type **integer**
 - Les flottants : le type **double**
 - Les tableaux : le type **array**
 - Les chaînes de caractères : le type **string**
 - Les objets
 - Rien : **NULL**
- Détermination du type de données
 - `gettype()`, `is_long()`, `is_double()`, `is_string()`, `is_array()`, `is_object()`, `is_bool()`

Les types de données (3/3)

Exemple:

Une variable de type chaîne de caractères est délimité par des simple ou doubles quotes.

```
<?php
$a_bool = TRUE; // un booléen
$a_str = "foo"; // une chaîne de caractères
$a_str2 = 'foo'; // une chaîne de caractères
$an_int = 12; // un entier

echo gettype($a_bool); // affiche : boolean
echo gettype($a_str); // affiche : string

// Si c'est un entier, incrément de 4
if (is_int($an_int)) {
    $an_int += 4;
}

// Si $a_bool est une chaîne de caractères, on l'affiche
if (is_string($a_bool)) {
    echo "String: $a_bool";
}
?>
```

conversion

- En PHP, on peut faire une conversion d'un type à un autre:

Var_finale2 = (nouveau-type) \$var_initiale


La concaténation

- Concaténer avec des guillemets doubles

```
1 <?php
2 $age_du_visiteur = 17;
3 echo "Le visiteur a $age_du_visiteur ans";
4 ?>
```

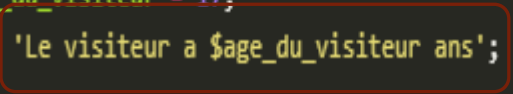
- Concaténer avec des guillemets simples

```
1 <?php
2 $age_du_visiteur = 17;
3 echo 'Le visiteur a ' . $age_du_visiteur . ' ans';
4 ?>
```



- Remarque

```
1 <?php
2 $age_du_visiteur = 17;
3 echo 'Le visiteur a $age_du_visiteur ans'; // Ne marche pas
4 ?>
```



Les opérateurs

- Les opérateurs
 - les opérateurs de calcul (+, -, /, %, *)
 - les opérateurs d'assignation (=)
 - les opérateurs d'incrémentation (++, --, +=, etc)
 - les opérateurs de comparaison (>, <, <=, ==, ===, etc)
 - les opérateurs logiques (and, or, &&, ||, etc)
 - Etc.

Les fonctions sur les chaines de caractères

Fonctions	Description
<i>strlen(\$str)</i>	Retourne le nombre de caractères d'une chaîne
<i>strtolower(\$str)</i>	Conversion en minuscules
<i>strtoupper(\$str)</i>	Conversion en majuscules
<i>str_word_count(\$str)</i>	Compte le nombre de mots utilisés dans une chaîne
<i>strrev(\$str)</i>	Inverse une chaîne
<i>trim(\$str)</i>	Suppression des espaces de début et de fin de chaîne
<i>substr(\$str,\$i,\$j)</i>	Retourne une sous-chaîne de <i>\$str</i> de taille <i>\$j</i> et débutant à la position <i>\$i</i>
<i>ord(\$char)</i>	Retourne la valeur ASCII du caractère \$char
....	

Les fonctions mathématiques

Fonctions	Description
<i>abs(\$x)</i>	valeur absolue
<i>ceil(\$x)</i>	arrondi supérieur
<i>floor(\$x)</i>	arrondi inférieur
<i>round(\$x,\$i)</i>	arrondi de x à la ième décimale
<i>pow(\$x,\$y)</i>	x exposant y
<i>max(\$a, \$b, \$c ...)</i>	retourne l'argument de valeur maximum
<i>rand()</i>	retourne un nombre aléatoire entre 0 et <i>getrandmax()</i>
<i>rand([\$x,\$y])</i>	retourne un nombre aléatoire entre \$x et \$y
....	

Les instructions conditionnelles (1/2)

- L'instruction **if**
 - **if** (condition réalisée) { liste d'instructions }
- L'instruction **if ... else**
 - **if** (condition réalisée) {liste d'instructions}
else { autre série d'instructions }
- L'instruction **if ... elseif ... else**
 - **if** (condition réalisée) {liste d'instructions}
elseif (autre condition) {autre série d'instructions }
else (dernière condition réalisée) { série d'instructions }
- Opérateur **ternaire**
 - (condition) ? instruction si vrai : instruction si faux;

Les instructions conditionnelles (2/2)

➤ L'instruction **switch**

```
switch (Variable) {  
    case Valeur1      : Liste d'instructions break;  
    case Valeur1      : Liste d'instructions break;  
    case Valeurs...   : Liste d'instructions break;  
    default           : Liste d'instructions break;  
}
```

Les instructions itératives (1/3)

► La boucle **for**

► **for** (expr1; expr2; expr3){instruction;}

► **for** (expr1; expr2; expr3): instruction; ...; **endfor**;

```
<?php for ($i=0; $i<=10; $i++): ?>
```

```
// Code en html
```

```
<?php endfor; ?>
```


Les instructions itératives (2/3)

- La boucle **while**
 - **while** (condition) {bloc d'instructions ;}
 - **while** (condition) :Instruction1 ;Instruction2 ;.... **endwhile;**

```
$i = 1;  
while ($i <= 10):  
    echo "$i ";  
    $i++;  
endwhile;
```

Les instructions itératives (3/3)

- La boucle **do...while**

- **do** {bloc d'instructions ;} **while** (condition) ;

- La boucle **foreach** (PHP₄)

- **foreach** (\$tableau **as** \$valeur) {insts utilisant \$valeur ;}

Les fonctions (1/3)

➡ Déclaration et appel d'une fonction

```
function nom_fonction ($arg1, $arg2, ...$argn)  
{  
    déclaration des variables ;  
    bloc d'instructions ;  
    //fin du corps de la fonction  
    return $resultat ;  
}
```

Les fonctions (2/3)

- ➡ Passage de paramètre **par référence**
 - ➡ Pour passer une variable par référence, il faut que son nom soit précédé du symbole **&** (exemple **&\$a**)

Les fonctions (3/3)

- Les fonctions anonymes (fermetures ou closures) permettent la création de fonctions sans préciser leur nom.

```
$a = 5;  
$b = 6;  
$som = function ($a, $b) {  
    return $a + $b;  
};  
echo $som($a, $b);
```

Les tableaux (1/5)

- Création à l'aide de la fonction **array()**
- Uniquement des tableaux à une dimension
 - Les éléments d'un tableau peuvent pointer vers d'autres tableaux
- Les éléments d'un tableau peuvent appartenir à des types distincts
- L'index d'un tableau en PHP commence de **0**
- La fonction **print_r()** pour afficher la structure d'un tableau

Les tableaux (2/5)

Fonctions	Description
<i>count()</i>	pour avoir le nombre d'éléments d'un tableau
<i>in_array(\$var,\$tab)</i>	dit si la valeur de \$var existe dans le tableau \$tab
<i>sort(\$tab)</i>	trie alphanumérique des éléments du tableau
<i>rsort(\$tab)</i>	trie alphanumérique inverse des éléments du tableau
<i>range(\$i,\$j)</i>	retourne un tableau contenant un intervalle de valeurs
<i>implode(\$str,\$tab)</i>	retournent une chaîne de caractères contenant les éléments du tableau \$tab joints par la chaîne de jointure \$str
<i>explode(\$delim,\$str)</i>	retourne un tableau dont les éléments résultent du hachage de la chaîne \$str par le délimiteur \$delim
<i>array_merge(\$tab1,\$tab2,\$tab3...)</i>	concatène les tableaux passés en arguments
<i>array_push(\$tab, \$val1 [, \$...])</i>	Ajoute une ou plusieurs valeurs au tableau \$tab
....	

Les tableaux (3/5)

Il y a deux types de tableau:

- ▀ les tableaux numérotés ou indexés ;
- ▀ les tableaux associatifs.

Les tableaux (4/5)

1. les tableaux numérotés ou indexés

- les valeurs sont rangées dans des « cases » différentes
- Chaque case est identifiée par un numéro, appelé **clé**.

Les tableaux (5/5)

➤ Exemple:

```
1 <?php
2 // La fonction array permet de créer un array
3 $prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');
4 ?>
```

1

```
1 <?php
2 $prenoms[0] = 'François';
3 $prenoms[1] = 'Michel';
4 $prenoms[2] = 'Nicole';
5 ?>
```

2

```
1 <?php
2 $prenoms[] = 'François'; // Créera $prenoms[0]
3 $prenoms[] = 'Michel'; // Créera $prenoms[1]
4 $prenoms[] = 'Nicole'; // Créera $prenoms[2]
5 ?>
```

3

Les tableaux (6/5)

- ➡ Parcourir un tableau numéroté:

```
1 <?php
2 // On crée notre array $prenoms
3 $prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoît');
4
5 // Puis on fait une boucle pour tout afficher :
6 for ($numero = 0; $numero < 5; $numero++)
7 {
8     echo $prenoms[$numero] . '<br />'; // affichera $prenoms[0], $prenoms[1] etc.
9 }
10 ?>
```

2. les tableaux associatifs

- ➡ appelé aussi **dictionnaire** ou **hashtable**
- ➡ Les éléments sont **référéncés** par des chaînes de caractères associatives en guise de nom: la *clé d'index*

Les tableaux (8/5)

➤ Exemple:

```
1 <?php
2 // On crée notre array $coordonnees
3 $coordonnees = array (
4     'prenom' => 'François',
5     'nom' => 'Dupont',
6     'adresse' => '3 Rue du Paradis',
7     'ville' => 'Marseille');
8 ?>
```

1

```
1 <?php
2 $coordonnees['prenom'] = 'François';
3 $coordonnees['nom'] = 'Dupont';
4 $coordonnees['adresse'] = '3 Rue du Paradis';
5 $coordonnees['ville'] = 'Marseille';
6 ?>
```

2

Les tableaux (9/5)

- ➡ Parcours d'un tableau associatif
- ➡ On doit utiliser la structure **foreach**

```
foreach ($personne as $elem)
{
    echo "$elem </br>";
}
```

```
foreach ($personne as $key => $elem)
{
    echo "$key : $elem";
}
```

```
1 <?php
2 // On crée notre array $coordonnees
3 $coordonnees = array (
4     'prenom' => 'François',
5     'nom' => 'Dupont',
6     'adresse' => '3 Rue du Paradis',
7     'ville' => 'Marseille');
8 ?>

1 <?php
2 $coordonnees['prenom'] = 'François';
3 $coordonnees['nom'] = 'Dupont';
4 $coordonnees['adresse'] = '3 Rue du Paradis';
5 $coordonnees['ville'] = 'Marseille';
6 ?>
```


Les tableaux (10/5)

➡ Exemple:

```
1 <?php
2 $coordonnees = array (
3     'prenom' => 'François',
4     'nom' => 'Dupont',
5     'adresse' => '3 Rue du Paradis',
6     'ville' => 'Marseille');
7
8 foreach($coordonnees as $element)
9 {
10     echo $element . '<br />';
11 }
12 ?>
```

François
Dupont
3 Rue du Paradis
Marseille

```
1 <?php
2 $coordonnees = array (
3     'prenom' => 'François',
4     'nom' => 'Dupont',
5     'adresse' => '3 Rue du Paradis',
6     'ville' => 'Marseille');
7
8 foreach($coordonnees as $cle => $element)
9 {
10     echo '[' . $cle . '] vaut ' . $element . '<br />';
11 }
12 ?>
```

[prenom] vaut François
[nom] vaut Dupont
[adresse] vaut 3 Rue du Paradis
[ville] vaut Marseille

Des questions