

# *Apprentissage et Pratique du Langage PHP*

## *Introduction:*

*Environnement PHP, Structures de  
données et de contrôle, Variables et  
fonctions, Bibliothèques, ... etc.*

*Par  
Elhassan Abdelwahed  
Département d'Informatique  
Faculté des Sciences Semlalia Marrakech*

# Programmation OO – Principes & Rappels

## Programmation Orientée Objets (OO)

Programmation basée sur la notion de **classes** (ensemble de classes d'**objets**)

### Apports :

Clarté (programmation centrée principalement sur les données, ... ),  
Modularité (encapsulation, description par classes, ... ),  
Réutilisation & Maintenance (héritage, polymorphisme, ... ),  
... etc.

**Langages** : Smalltalk, C++, Java, ..., PHP (à partir de sa version 5)

# PHP – Origines & Historique

---



**1995-PHP/FI** (*Personnal Home Page/Interpréteur de formulaire*): est un langage de scripts développé (en Perl ensuite en C) par Rasmus Lerdorf (<http://lerdorf.com>) pour des besoins de pages web personnelles: Compteur, livre d'or, ...etc. Rasmus décide de proposer son code à la communauté

**1997- PHP 3** (*Hypertext Preprocessor*): est une refonte de PHP/FI 2.0: l'analyseur fut de nouveau réécrit par Zeev Suraski et Andi Gutmans (les fondateurs de ZEND) . PHP 3.0 est stable, fonctionnel et extensible: avec la possibilité d'y intégrer des nouveaux modules réalisés par la communauté PHP (modules: images dynamique, génération de PDF, ... etc.). En 1998, PHP 3.0 était installé sur 10% du parc mondial des serveurs web.

**2000- PHP 4** : Dans un souci de montée en performance, Zeev et Andi développent le moteur *Zend Engine*. En 2000, ce moteur fut intégré à PHP sous la forme de sa version PHP 4.0 avec plus de nouvelles fonctionnalités: sessions, tampon de sorties, sécurité, ... etc.

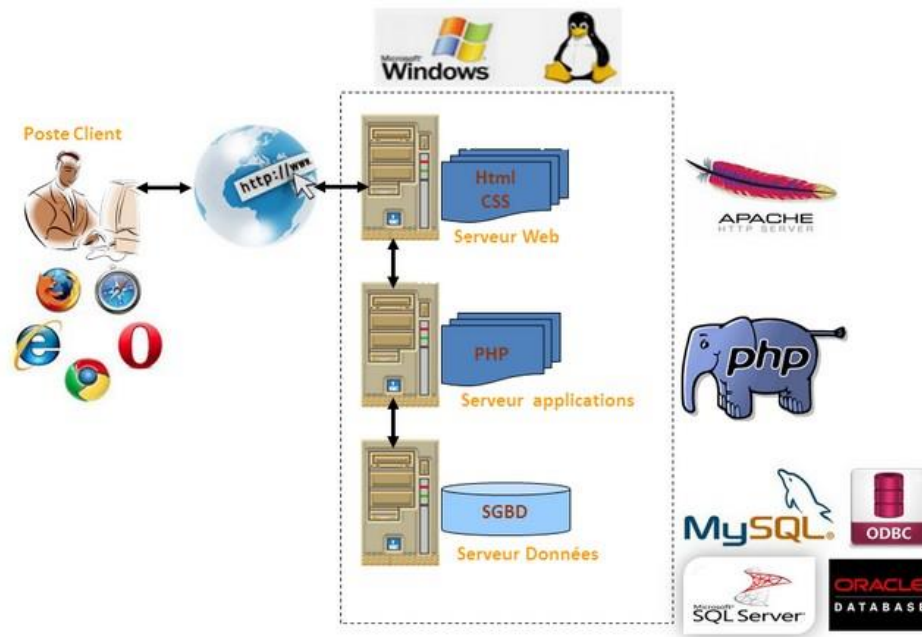
**A partir de 2005- PHP 5** : les travaux d'extension de PHP en repris pour une version plus professionnelle et plus simple. La version stable, basée sur *Zend Engine 2* , est sortie en 2005: POO, gestion des erreurs, prise en charge de XML, PDO, Service Web, SQLite, ... etc.

# PHP – Origines & Historique

PHP est un langage basé sur les scripts incorporés dans du HTML et exécutés côté serveur. La syntaxe de PHP dérive de C et de Perl (attention à la case !)

But(s) de PHP: Développement d'applications Web (sites ou autres) dynamiques, interactives, modulaires et interopérables.

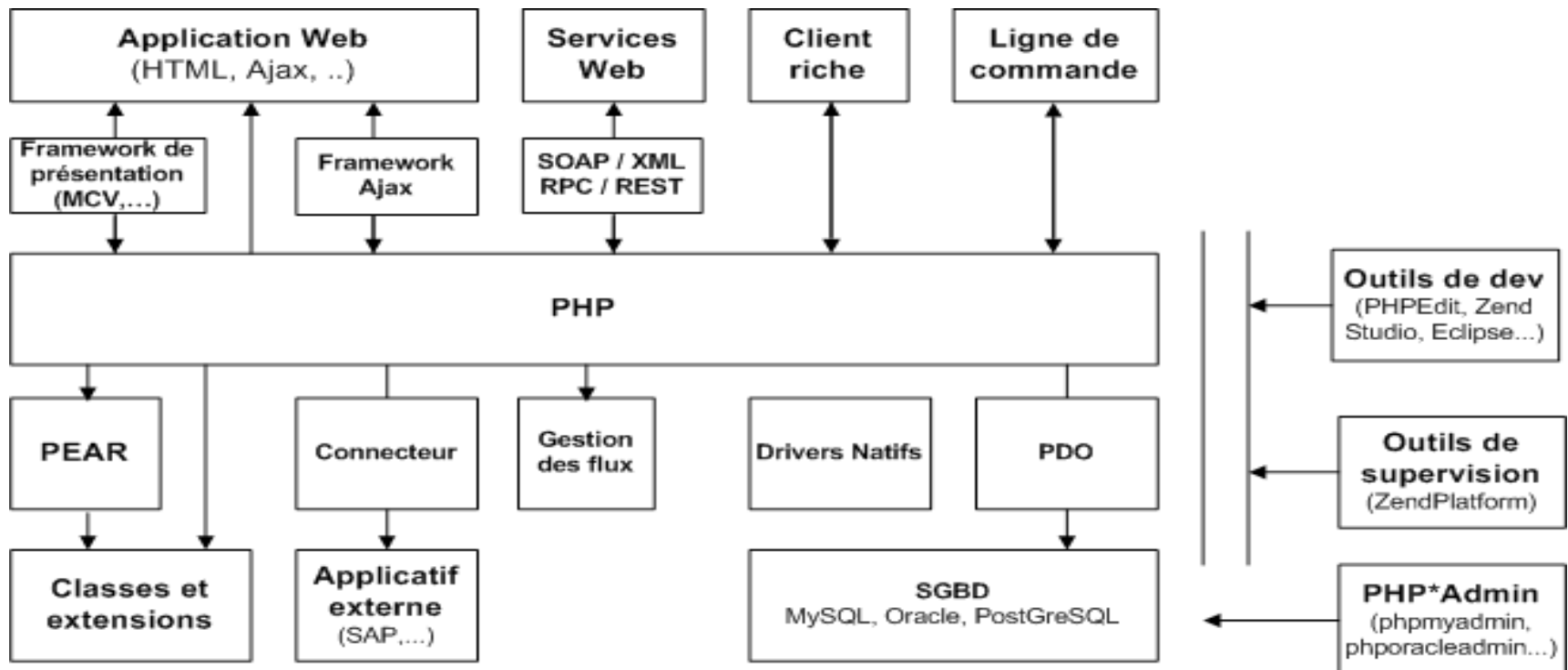
A partir de la version 5, PHP est devenue une solution reconnue comme viable (stabilité, fiabilité, performance). PHP 5 a été propulsé parmi les plates-formes d'entreprises comme J2EE ou .Net:



# PHP – Origines & Historique

PHP est un langage: ouvert, gratuit, extensible, portable, ... etc.

PHP possède une grande quantité d'outils (modules, Bibliothèques): Manipulation d'images, gestion des fichiers (PDF, XML, etc.) , bases de données (MySQL, Oracle, PostgreSQL, SQL Server, Sybase, Empress, Informix, etc.), prise en compte de protocoles (TCP, SMTP, LDAP, SOAP, etc.), Connecteurs applicatifs (SAP, Lotus Notes, etc.), ... etc.

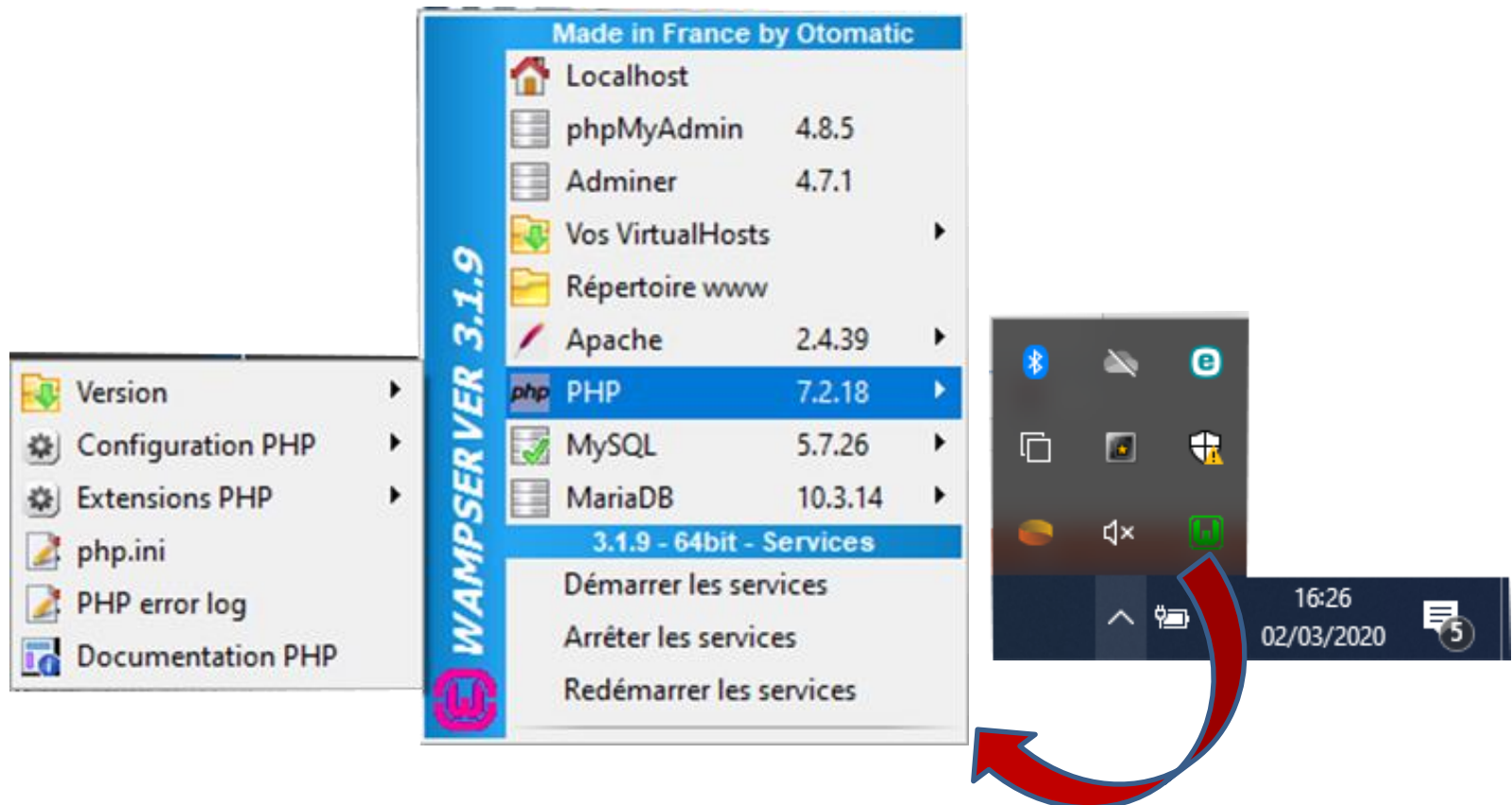


# PHP – Installation



<http://www.easyphp.org/>

<http://www.wampserver.com/>





# HTML - PHP

Un programme (script(s)) PHP est soit intégré dans un fichier HTML ou sauvegarder dans un fichier externe (.php) et invoqué à partir d'un fichier HTML.

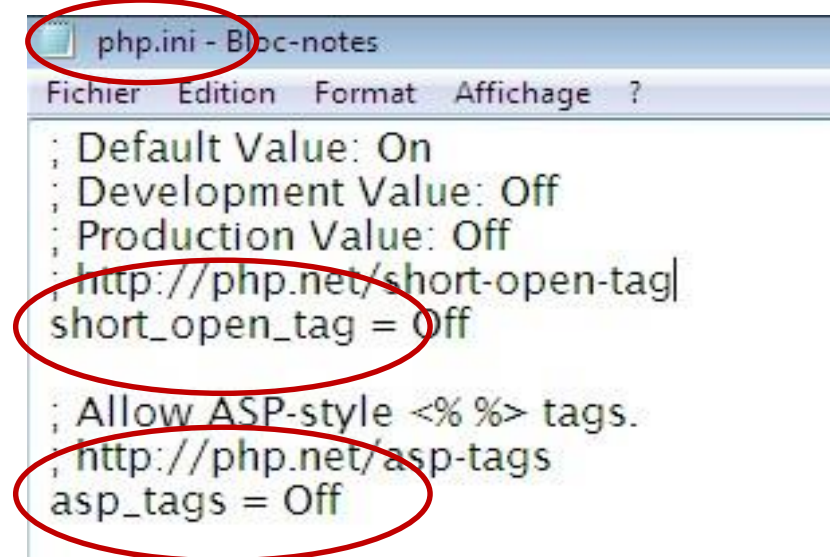
Il existe quatre balises pour incorporer PHP dans du HTML:

`<? ?>`

`<?php ?>`

`<script language= "php"> </script >`

`<% %>`



# HTML - PHP

```
<html>
  <head> <title> Exemple 1 en PHP </title> </head>
  <body>
    <h1> Département d'Informatique FSSM </h1>
    <?php
      echo "<h1> Formations: Licence & Master </h1>";
    ?>
  </body>
</html>
```

← → ↻ ⓘ localhost/dossier/exemple1.php

**Département d'Informatique FSSM**

**Formations: Licence & Master**



Le moteur d'interprétation du langage lit un fichier source PHP: Les instructions PHP n'apparaissent pas dans le résultat généré et envoyé au navigateur.



# Structure du langage PHP

**PHP ressemble aux langages C, C++ et Javascript**

- **(;) à la fin de chaque ligne d'instructions**
- **{...} pour encadrer un bloc d'instructions**
- **les opérateurs de comparaison et d'affectation: &&, ||, ==, ...**
- **Les symboles des commentaires: // et /\* ... \*/**
- **Structures de contrôle:                                   for( ; ; ; ) { ... }, while { ... }, etc.**
- **Comme en C, PHP support les commentaires:**
  - /\* Ceci est un commentaire \*/**
  - // Ceci est un commentaire de fin de ligne**
- ... etc.**

# Les variables

- Les identificateurs de variable sont précédés du symbole \$
- Le typage des variables est implicite en PHP: En fonction du contexte dans lequel la variable est utilisée, PHP détermine son type au moment de l'exécution du programme<sup>(\*)</sup>.
- PHP peut changer dynamiquement le type de variable selon son contenu.
- Les variables peuvent être de type : entier (integer), réel (float, double), chaîne de caractères (string), tableau (array), booléen (boolean), etc.
- **bool settype ( \$var , type )**: Déclaration explicite du type d'une variable

```
<html>
  <head> <title> Variables en PHP </title> </head>
  <body>
    <?php
      $salaire; $age= 22; $nom = "Slimani";
      $varA = "5bar";      // chaîne
      $varB = true;        // booléen
      settype($varA, "integer"); // $varA vaut maintenant 5 (integer)
      settype($varB, "string");  // $varB vaut maintenant "1" (string)
      echo "<h1> Mr $nom a $age ans </h1>";
      echo "<h1> varA: $varA et varB: $varB </h1>";
    ?>
  </body>
</html>
```

← → ↻ ⓘ localhost/dossier/exemple2.php

**Mr Slimani a 22 ans**

**varA: 5 et varB: 1**

# Opération sur les variables

## Opérateurs

<b>Arithmétiques:</b>	<b>+ ; - ; * ; / ; % ; ++ ; -- ;</b>
<b>Assignements:</b>	<b>= ; -= ; *= ; /= ; %=</b>
<b>Logiques:</b>	<b>and ou &amp;&amp; ; or ou    ; !</b>
<b>Comparaisons:</b>	<b>== ; != ; &lt;= ; &lt; ; &gt;= ; &gt;</b>

**\$a = (\$b=4) + 5; // \$a vaut 9 et \$b vaut 4**  
**\$s = "Bonjour"; \$s .= "Tout le monde";**

# Exemple de fonctions utiles sur les variables

Fonctions	Commentaires
gettype (\$nom_var)	détermine le type de données de la variable
settype (\$nom_var, "type")	définit explicitement le type de variable
isset (\$nom_var)	sert à savoir si une variable possède une valeur (true ou false)
unset (\$nom_var)	détruit une variable
empty (\$nom_var)	renvoie true si la variable est vide ou vaut 0, sinon renvoie false
is_int (\$nom_var) is_integer (\$nom_var) is_long(\$nom_var)	détermine si la variable est un entier
is_double (\$nom_var) is_float (\$nom_var) is_real (\$nom_var)	détermine si la variable est un double
is_string (\$nom_var)	détermine si la variable est une chaîne
is_array (\$nom_var)	détermine si la variable est un tableau
is_object (\$nom_var)	détermine si la variable est un objet
intval (\$nom_var)	définit un entier
doubleval (\$nom_var)	définit un double
strval (\$nom_var)	définit une chaîne de caractères

# Variables d'environnement

Les variables d'environnement sont des informations concernant l'environnement du script au niveau du serveur et du client (type de serveur, le chemin et le nom du script appelé, le navigateur du client, etc.). Ces variables sont stockées directement dans les variables associées à leur nom.

Pour y accéder : superglobal `$_SERVER` ou à l'aide de la fonction `getenv`

```
<?
echo '<p>Avec $_SERVER : ' . $_SERVER['HTTP_USER_AGENT'] . '</p>';
echo '<p>Avec getenv() : ' . getenv('HTTP_USER_AGENT') . '</p>';
?>
```

Nom	Description (exemple)
DOCUMENT_ROOT	Racine du serveur (/home/www/phpfrance)
HTTP_ACCEPT_LANGUAGE	Langage accepté par le navigateur client (fr,ie-ee;q=0.5)
HTTP_HOST	Nom de domaine du serveur (phpfrance.com)
HTTP_USER_AGENT	Navigateur (et système) et client (Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.7.5) Gecko/20041109 Firefox/1.0)
REMOTE_ADDR	Adresse IP du client (212.78.54.36)
REMOTE_PORT	Port sur lequel la requête HTTP a été envoyée au serveur (1211)
SERVER_ADDR	Adresse IP du serveur (209.15.23.241)
SERVER_ADMIN	Adresse de l'administrateur du serveur (damien@phpfrance.com)
SERVER_NAME	Nom local du serveur (localhost)
SERVER_SIGNATURE	Type de serveur (Apache/1.3.12 Server at 127.0.0.1 Port 80)
REQUEST_METHOD	Méthode d'appel du script (GET)
QUERY_STRING	Liste des paramètres passés au script (id=14&page=3&action=voir)
REQUEST_URI	Chemin du script (/chemin/script.php?id=14&page=3&action=voir)
PATH_INFO	Chemin web du script (/chemin/script.php)
PATH_TRANSLATED	Chemin complet du script (/home/www/phpfrance/chemin/script.php)

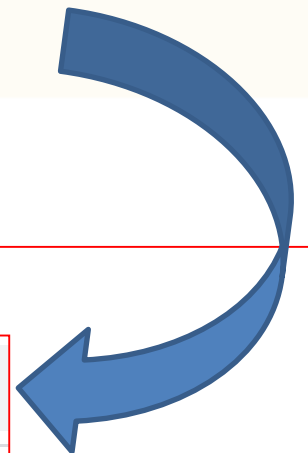
# Constante, variable statique et référence

```
<html>
<head> <title> Constantes, Variable statique et Référence en PHP </title> </head>
<body>
    <?php
        define("FILIERE", "Smi Informatique"); // Définition d'une Constante
        echo FILIERE;
        const ETABLISSENT = 'Fssm'; // Autre méthode pour définir une Constante
        static $semestre = 'S1'; // variable statique
        echo "    Semestre $semestre ";
        $a = 100;      $b = &$a;      // la variable $b fait référence à $a
        $a++;          // $b vaut alors 101
        echo " a: $a    b: $b ";
    ?>
</body>
</html>
```



localhost/dossier/exemple3.php

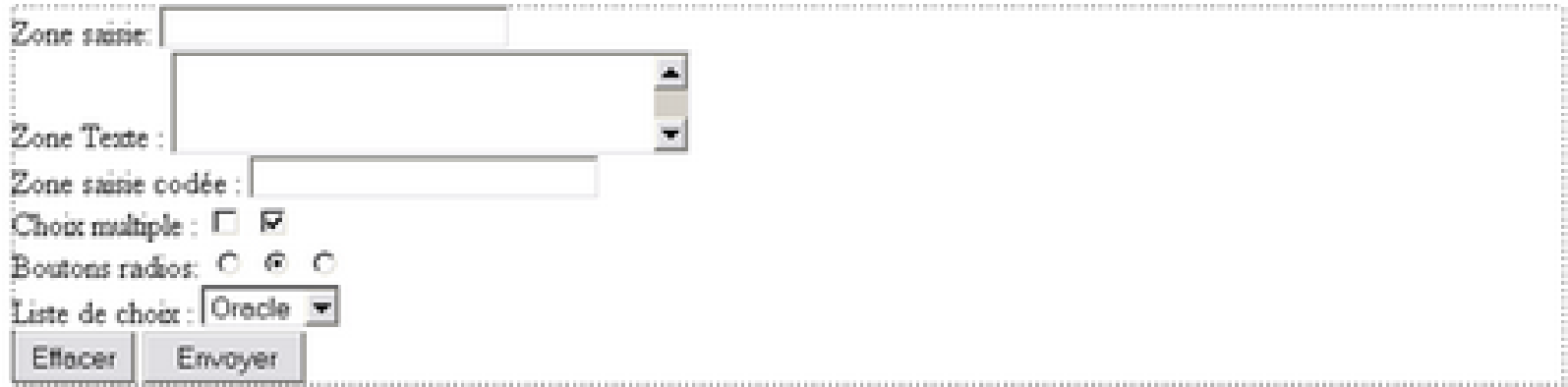
Smi Informatique Semestre S1 a: 101 b: 101





# Entrée - Sortie

**En général, les saisies de données se fait à travers les éléments (zone de saisie, etc.) d'un des formulaires de l'interface utilisateur.**



The image shows a web form with the following elements:

- Zone saisie:** A text input field.
- Zone Texte :** A larger text area with a vertical scrollbar.
- Zone saisie codée :** A text input field.
- Choix multiple :** Two checkboxes, the second of which is checked.
- Boutons radios:** Three radio buttons, the middle one is selected.
- Liste de choix :** A dropdown menu currently showing 'Oracle'.
- Buttons:** Two buttons at the bottom labeled 'Effacer' and 'Envoyer'.

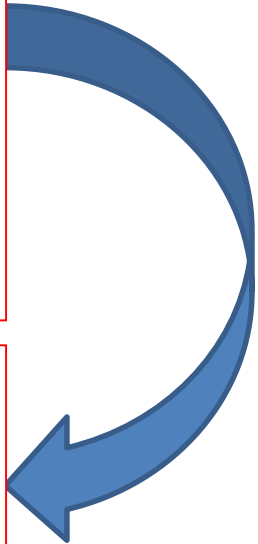
**Les édition des résultats (sorties) vont s'inscrire dans la page par les fonctions :**

```
echo ("message");  
print ("message");
```

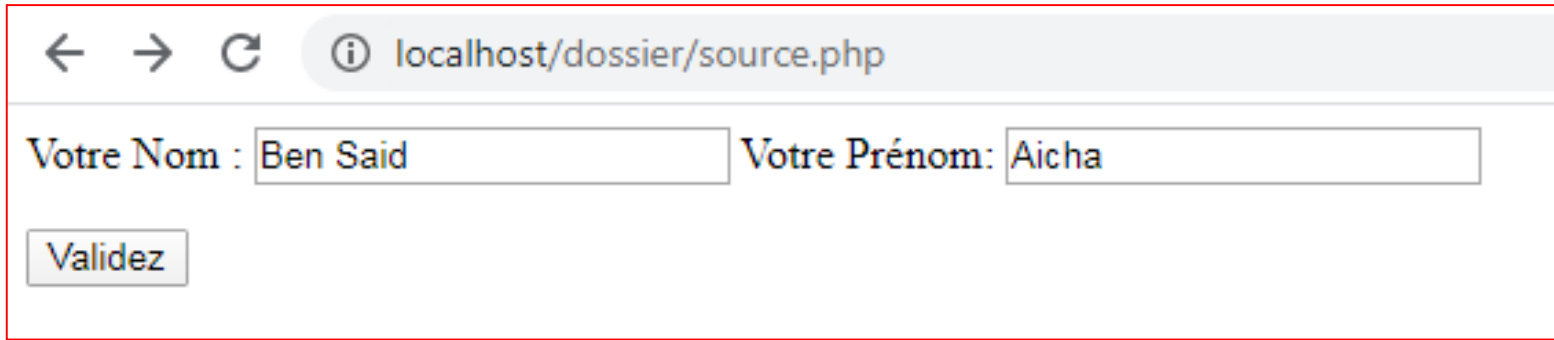
# Variables Issues des formulaires: GET

```
<html>
  <head> <title> Formulaire en PHP </title> </head>
  <body>
    <form action="destination.php" method = "get" >
      <p>
        Votre Nom    : <input type="text" name="nom">
        Votre Prénom: <input type="text" name="prenom">
      </p>
      <p> <input type="submit" value= "Validez"> </p>
    </form>
  </body>
</html>
```

```
<html>
  <head> <title> Formulaire en PHP </title> </head>
  <body>
    <?php
      $n = $_GET["nom"];   $p = $_GET["prenom"];
      echo "<h1> Bonjour $p $n </h1>";
    ?>
    <p> <a href="source.php"> Retour au formulaire </a> </p>
  </body>
</html>
```

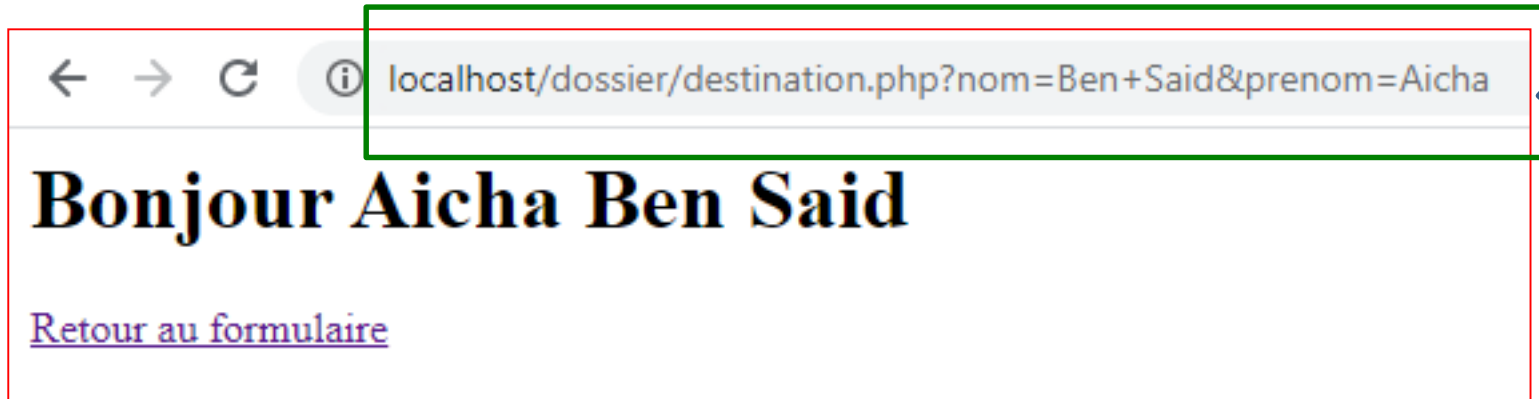


# Variables Issues des formulaires: GET



← → ↻ ⓘ localhost/dossier/source.php

Votre Nom :  Votre Prénom:



← → ↻ ⓘ localhost/dossier/destination.php?nom=Ben+Said&prenom=Aicha

**Bonjour Aicha Ben Said**

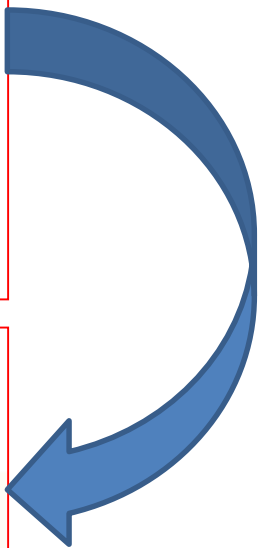
[Retour au formulaire](#)

Les variables saisies dans le formulaire sont transmises au programme (script) PHP dans l'URL et elles sont récupérées dans le tableau global `$_GET[ ]`

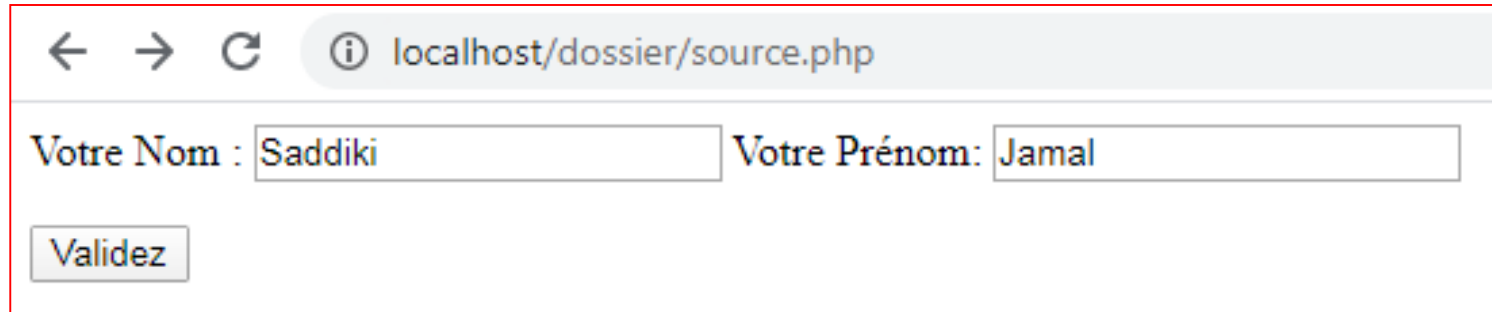
# Variables Issues des formulaires: POST

```
<html>
  <head> <title> Formulaire en PHP </title> </head>
  <body>
    <form action="destination.php" method = "post" >
      <p>
        Votre Nom    : <input type="text" name="nom">
        Votre Prénom: <input type="text" name="prenom">
      </p>
      <p> <input type="submit" value= "Validez"> </p>
    </form>
  </body>
</html>
```

```
<html>
  <head> <title> Formulaire en PHP </title> </head>
  <body>
    <?php
      $n = $_POST["nom"];   $p = $_POST["prenom"];
      echo "<h1> Bonjour $p $n </h1>";
    ?>
    <p> <a href="source.php"> Retour au formulaire </a> </p>
  </body>
</html>
```



# Variables Issues des formulaires: POST



← → ↻ ⓘ localhost/dossier/source.php

Votre Nom :  Votre Prénom:



← → ↻ ⓘ localhost/dossier/destination.php

**Bonjour Jamal Saddiki**

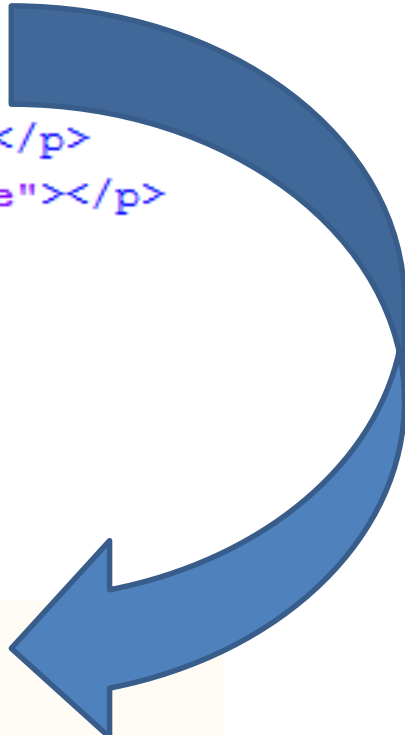
[Retour au formulaire](#)

Les variables saisies dans le formulaire sont transmises au programme (script) PHP dans le tableau global `$_POST[ ]`

# Variables Issues des formulaires

```
<html>    <head>    <title>Formulaire en PHP</title>  </head>
<body>    <form action="go.php" method="post">
  <p>Votre nom : <input type="text" name="nom"></p>
  <p>Votre prénom : <input type="text" name="prenom"></p>
  <p><input type="submit" value="envoyer le formulaire"></p>
  </form>  </body>
</html>
```

```
<html>
<head>    <title>Formulaire en PHP</title>  </head>
<body>    <?php
  $n = $_POST['nom'];  $p = $_POST['prenom'];
  echo "<h1> Bienvenue à $p $n </h1>";
  ?>
  <p><a href="frm.php">Retour au formulaire</a></p>  </body> </html>
```





# Les fonctions

**Les fonctions peuvent prendre des arguments dont il n'est pas besoin de spécifier le type. Elles peuvent de façon optionnelle retourner une valeur.**

```
<html>
  <head> <title> Constantes, Variable statique et Référence en PHP </title> </head>
  <body>
    <?php
      // Définition de fonction avec des arguments ayant des valeurs par défaut
      function PrixTTC ( $PrixHT, $Tva = 0.1 )
      {
          $spr = $PrixHT + $PrixHT * $Tva ;
          return ( $spr);
      }

      $PrixTV = PrixTTC (100); // Avec $PrixHT = 100, $Tva = 0.1 et $PrixTV = 110
      $PrixLux = PrixTTC( 1000, 0.3); // Avec $PrixHT = 1000, $Tva = 0.3 et $PrixLux = 1300

      // Définition de fonction avec passage d'arguments par référence
      function ParReference ( &$var ) { $var++; }
      $a=5;
      ParReference ($a); // $a vaut 6
    ?>
  </body>
</html>
```

# Portée des variables

La portée d'une variable définie dans une fonction est locale à cette fonction.

Une variable globale, est définie tout au début du script, en dehors et avant toute fonction. Elle est déclarée à l'aide du mot-clé `global` au sein des fonctions.

`$GLOBALS` contient des références sur toutes les variables globales actuellement définies.

```
<?php
$annee = "2002";
$prenom = "Victor";
$nom = "Hugo";
function bicentenaire()
{
    global $annee, $prenom, $nom;
    echo "$annee c'était le bicentenaire de la naissance de $prenom $nom !";
    $annee = "2003";
}
bicentenaire("bicentenaire");
// affiche "2002 c'était le bicentenaire de la naissance de Victor Hugo!"
echo $annee; // affiche 2003
?>
```

```
<?php
$a=1;  $b=2;
function somme( ) { $GLOBALS['b'] = $GLOBALS['a'] + $GLOBALS['b']; }
somme( ); echo $b;
?>
```

# Les tableaux

## Tableaux

Une variable tableau est de type array. Un tableau accepte des éléments de tout type. Les éléments d'un tableau peuvent être de types différents et sont séparés d'une virgule. L'appel d'un élément du tableau se fait à partir de son indice dont l'origine est zéro.

## Exemples

```
$Couleurs = array('red', 'yellow', 'blue', 'white');  
$Mixte = array('SMI', 2019, 14.5, $Sem);  
$prenoms[ ] = "Said"; $prenoms[ ] = "Fatima"; $prenoms[ ] = "Omar";  
$villes[0] = "Paris"; $villes[1] = "Londres"; $villes[2] = "Lisbonne";
```

## Modification des indices

```
$Modules = array (1 => "Java", "PHP", "Oracle") ; // Modules[2]= "PHP"  
$Module = array ("Java", "PHP", 5 => "UML") ; // Module[0]="Java" , Module[5]="UML"
```

## Tableau associatif

```
$TA["a"] = "EA"; $ TA["b"] = "EB"; $ TA["c"] = "EC" ;  
ou  
$TA = array ("a" => "EA", "b" => "EB« , "c" => "EC" );
```

## Tableau multidimensionnel

```
$Tab[0] [0]= "Casa"; $Tab [0] [1]= "Agadir";  
$Tab [1] [0]= "Tanger"; $Tab[1] [1]= "Fez";  
ou  
$Tab = array ( array ("Casa", "Agadir"), array ("Tanger", "Fez") );
```

# Fonctions relatives aux tableaux

**count(\$tab), sizeof**

Retournent le nombre d'éléments du tableau

**in\_array(\$var,\$tab)**

Vérifie si la valeur de \$var existe dans le tableau \$tab

**list(\$var1,\$var2...)**

Transforme une liste de variables en tableau

**range(\$i,\$j)**

Retourne un tableau contenant un intervalle de valeurs

**shuffle(\$tab)**

mélange les éléments d'un tableau

**sort(\$tab)**

Trie alphanumérique les éléments du tableau

**rsort(\$tab)**

trie alphanumérique inverse les éléments du tableau

**implode(\$str,\$tab)**

Retourne une chaîne contenant les éléments du tableau \$tab joints par la chaîne \$str

**explode(\$delim,\$str)**

Retourne un tableau résultat du hachage de la chaîne \$str par le délimiteur \$delim

**array\_merge(\$tab1,\$tab2,\$tab3...)**

concatène les tableaux passés en arguments

**array\_rand(\$tab)**

retourne un élément du tableau au hasard

# Chaînes de caractères

**Les chaînes de caractères peuvent être définies en utilisant deux type de délimiteurs:    double guillemets ,   simple guillemets**

**Les chaînes peuvent concaténée par l'opérateur (.)**

**\$d= "Département" ; \$i="d\ 'informatique"; \$di = \$d . \$i**

**Quelques fonctions:**

**strlen(\$str)** : retourne le nombre de caractères d'une chaîne

**strtolower(\$str)** : conversion en minuscules

**strtoupper(\$str)** : conversion en majuscules

**trim(\$str)** : suppression des espaces de début et de fin de chaîne

**substr(\$str,\$i,\$j)** : retourne une sous chaîne de \$str de taille \$j et débutant à la position \$i

**strnatcmp(\$str1,\$str2)** : comparaison de 2 chaînes

**ord(\$char)** : retourne la valeur ASCII du caractère \$char

**Les fonctions d'affichage :**

**echo()** : Écriture dans le navigateur

**print()** : Écriture dans le navigateur

**printf([\$format, \$arg1, \$arg2])** : Affichage formaté

**echo "Bonjour \$name";**

**print("Bonjour \$name");**

**printf("Bonjour %s", \$name );**

# Fonctions Mathématiques

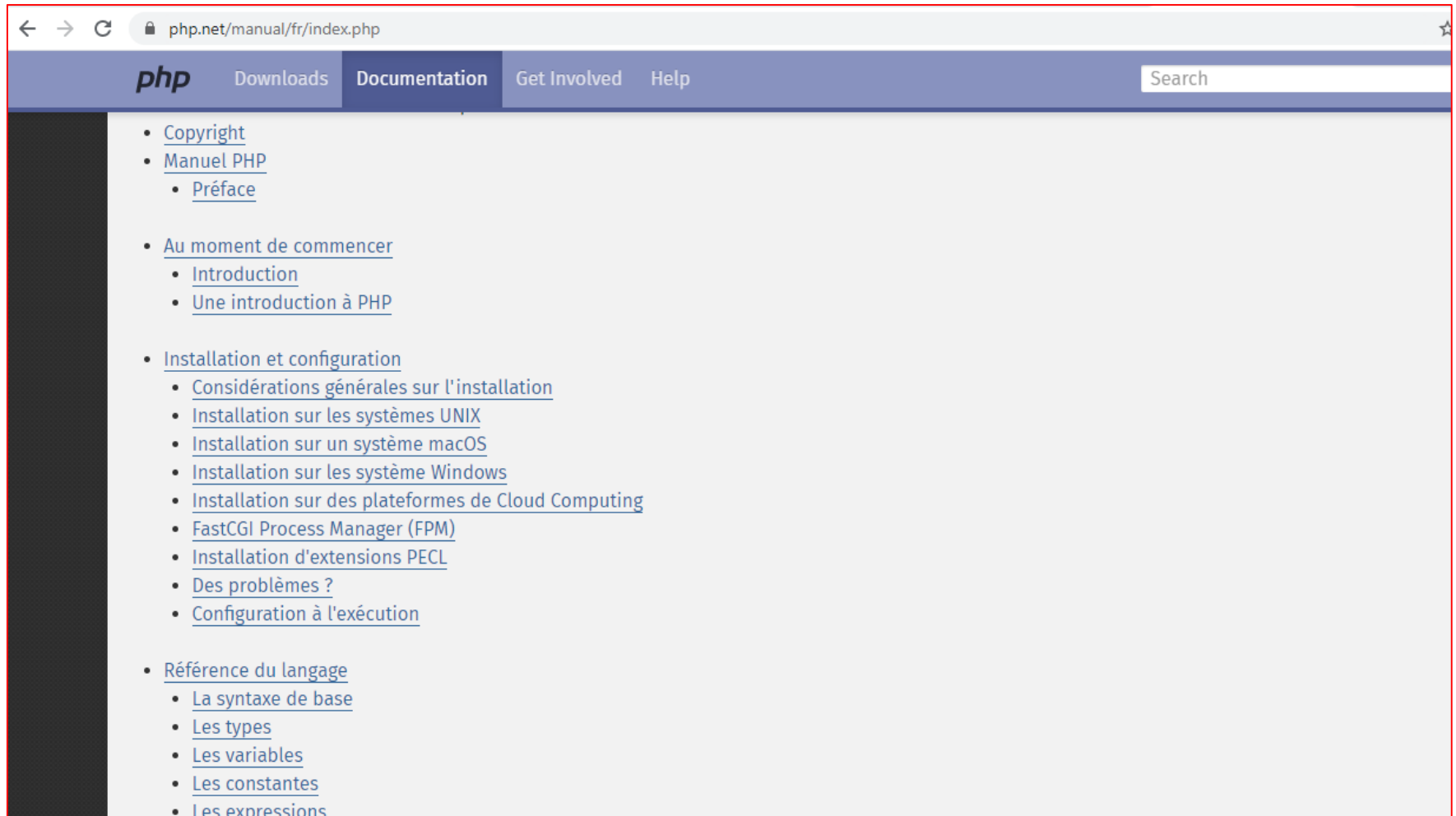
<b>abs(\$x) :</b>	<b>valeur absolue</b>
<b>ceil(\$x) :</b>	<b>arrondi supérieur</b>
<b>floor(\$x) :</b>	<b>arrondi inférieur</b>
<b>pow(\$x,\$y) :</b>	<b>x exposant y</b>
<b>round(\$x,\$i) :</b>	<b>arrondi de x à la ième décimale</b>
<b>max(\$a, \$b, \$c) :</b>	<b>retourne l'argument de valeur maximum</b>
<b>Fonctions trigo.:</b>	<b>cos ( \$x ), sin ( \$x ), tan ( \$x ), ...</b>
<b>Autres:</b>	<b>exp ( \$x ),, log ( \$x ), ...</b>
<b>... etc.</b>	



# Manuel officiel de PHP

Pour plus d'informations (fonctions, ...etc.) Consultez le manuel officiel de PHP :

<https://www.php.net/manual/fr/index.php>



## Objectifs de l'atelier 2

- Installation et configuration de l'environnement de développement PHP
- Développement de programme PHP (structures de données et de contrôle, variables, fonctions, ... etc.) sur un projet de gestion des notes
- Le programme de gestion des notes va évoluer (et s'améliorer) avec notre avancement dans les différents chapitres en y intégrant à chaque nouveau chapitre les différentes notions étudiées.