

Développement Web (R112)

MMI, IUT1, UGA

Gwen Salaün

PHP – Bases pour la programmation

(cours 5)

Plan

- Premiers pas avec PHP
- Les variables et l'affectation
- Les tableaux
- Lecture/écriture
- Les structures de contrôle
- Les fonctions

PHP

Un langage pour créer des sites web dynamiques, *i.e.*, qui peuvent changer sans intervention humaine

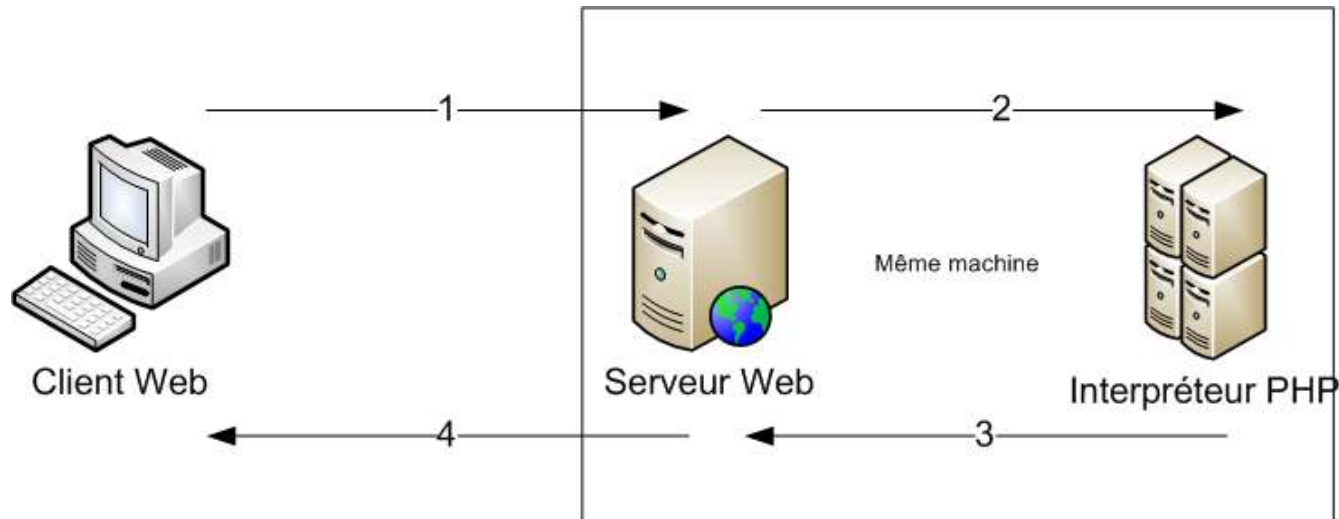
Mais aussi un langage :

- Orienté objet
- Interprété
- Typé à l'exécution
- Exécutable en ligne de commande ou côté serveur

Les balises PHP

- Les pages web contenant du PHP ont l'extension '.php'
- Une page PHP est en fait une simple page HTML qui contient des instructions en langage PHP
- Les instructions PHP sont placées dans une balise
`<?php /* code PHP */ ?>`
- Il est possible d'ajouter des commentaires en PHP pour décrire le fonctionnement du code. On utilise pour cela les symboles `// ..` ou `/* .. */`
- Pour afficher du texte on utilise l'instruction 'echo'

PHP en mode client/serveur



1. Le client demande la ressource
2. Le serveur Web identifie le type de la ressource en fonction de son extension (.php)
3. L'interpréteur fabrique la ressource (la page, une image, ...) en reconnaissant les balises `<?php /* code */ ?>`
4. Le serveur Web renvoie la ressource

PHP en R112

- Dans ce module, on ne fait pas (encore) de web
- On utilise PHP comme un langage de programmation classique (même si on visualise les résultats dans une page web..)
- Pourquoi utiliser PHP ? Car on (ré-)utilisera PHP au S2 pour faire du web dynamique

Plan

- Premiers pas avec PHP
- **Les variables et l'affectation**
- Les tableaux
- Lecture/écriture
- Les structures de contrôle
- Les fonctions

Déclaration de variable

- Variables déclarées en utilisant \$
- `isset()` permet de savoir si une variable est définie
- Deux principaux affichages :
 - `echo` pour afficher la valeur d'une expression
 - `var_dump` pour un affichage de débogage

```
<?php
    $a;                // Déclaration de $a
    var_dump($a);       // Notice: Undefined variable: a
    var_dump(isset($a)); // Affichage pour débogage de $a
    echo $a;           // Notice: Undefined variable: a
?>
```

Affectation et typage

- L'affectation s'écrit en utilisant le symbole '='
- Les variables sont typées, mais pas de typage explicite, et le type peut changer à l'exécution

```
<?php
    $a = 0;                // a est un entier et vaut 0
    var_dump($a);
    $a = true;             // a est un booléen et vaut true
    var_dump($a);
    $a = $a + 1;           // a est converti en entier, vaut 1
                           // et on lui ajoute 1
                           // a est un entier qui vaut 2

    var_dump($a);
?>
```

Recopie de variable

- Un langage par recopie que se soit pour les types primitifs, tableaux ou objets.
- La référence peut-être forcées avec &

```
<pre>
<?php
    $a = "12";
    $b = $a;
    $c = &$a;          // $a et $c partagent le même espace mémoire
    $a = "12"."34";    // . est l'opérateur de concaténation
    var_dump($a);      // 1234
    var_dump($b);      // 12
    var_dump($c);      // 1234
?>
</pre>
```

Transtypage

- Possibilité de changer explicitement de type, transtypage (type).

```
<?php
    $a = true;
    var_dump($a);    // $a est un booléen et vaut true
    $a = (int) $a;
    var_dump($a);    // $a est un entier et vaut 1
?>
```

Constantes

- Une constante est une variable dont la valeur ne change pas
- Utilisation du mot clé 'define' : define(nom, valeur)

```
<?php
```

```
define("NOM","Dupont");  
echo NOM;
```

```
define("PRENOM","Jean");  
echo PRENOM;
```

```
define("NOM_COMPLET", NOM . " " . PRENOM);  
echo NOM_COMPLET;
```

```
?>
```

Plan

- Premiers pas avec PHP
- Les variables et l'affectation
- **Les tableaux**
- Lecture/écriture
- Les structures de contrôle
- Les fonctions

Les tableaux (1)

- Un tableau consiste en un ensemble d'éléments ordonnés accessibles par leur indice ou clé
- Les tableaux sont associatifs et de taille variable : les clés sont libres

```
<?php
```

```
$tab = array();           // $tab est un tableau vide  
var_dump($tab);  
$tab["un"]="avion";       // insertion d'avion sous la clé un  
$tab["deux"]="velo";       // insertion de velo sous la clé deux  
$tab["trois"]="camion";    // insertion de camion sous la clé trois
```

```
$tab["un"]="moto";        // insertion de moto sous la clé un  
var_dump($tab);
```

```
?>
```

moto	velo	camion
un	deux	trois

Les tableaux (2)

```
<?php
$tab2 = array(
    "un"      => "moto",
    "deux"    => "velo",
    0         => "auto",
    2         => "camion"
);
var_dump($tab2);
?>
```


Les tableaux (3)

- Des tableaux de tableaux

```
$annuaire = array (  
    0 => array (  
        "nom"      => "durand",  
        "prenom"   => "paul",  
        "tel"      => "06 00 00 01"  
    ),  
    1 => array (  
        "nom"      => "dupond",  
        "prenom"   => "jean",  
        "tel"      => "06 00 00 02"  
    )  
);
```

Plan

- Premiers pas avec PHP
- Les variables et l'affectation
- Les tableaux
- **Lecture/écriture**
- Les structures de contrôle
- Les fonctions

Lecture

- PHP est un langage web et le moyen habituel de lire de l'information est d'utiliser un **formulaire** (ils seront étudiés au S2)
- Dans ce cours, pour simplifier, nous écrirons directement les valeurs dans des variables pour simuler la lecture

// Pseudo-code

$x \leftarrow \text{Lire}()$

// on tape 3 au clavier



// PHP

$\$x = 3;$

Écriture avec l'instruction 'echo'

- Elle permet d'insérer du texte dans la page web
- Le texte peut contenir des balises HTML :

echo "Ceci est du texte" ;

- On peut aussi afficher plusieurs lignes en même temps avec la commande suivante :

echo <<<END

Mon texte

est vraiment

trop long !

END;

Plan

- Premiers pas avec PHP
- Les variables et l'affectation
- Les tableaux
- Lecture/écriture
- **Les structures de contrôle**
- Les fonctions

La structure conditionnelle 'if'

```
<?php
    if (cond) {
        ...
    } else {
        ...
    }
?>
```

Un exemple :

```
1 <?php
2 $age = 8;
3
4 if ($age <= 12)
5 {
6     echo "Salut gamin !";
7 }
8 ?>
```

Symboles de comparaison

Quelques symboles à connaître car très utiles pour écrire des conditions :

Symbole	Signification
==	Est égal à
>	Est supérieur à
<	Est inférieur à
>=	Est supérieur ou égal à
<=	Est inférieur ou égal à
!=	Est différent de

Attention : '==' permet de tester l'égalité alors que '=' est utilisé pour affecter une valeur à une variable !

Un exemple de 'if .. else'

```
1 <?php
2 $age = 8;
3
4 if ($age <= 12) // SI l'âge est inférieur ou égal à 12
5 {
6     echo "Salut gamin ! Bienvenue sur mon site !<br />";
7     $autorisation_entrer = "Oui";
8 }
9 else // SINON
10 {
11     echo "Ceci est un site pour enfants, vous êtes trop vieux pour pouvoir entrer.
12     Au revoir !<br />";
13     $autorisation_entrer = "Non";
14 }
15 echo "Avez-vous l'autorisation d'entrer ? La réponse est : $autorisation_entrer";
16 ?>
```


Un autre exemple de 'if .. else'

```
1 <?php
2 if ($autorisation_entrer)
3 {
4     echo "Bienvenue petit nouveau. :o)";
5 }
6 else
7 {
8     echo "T'as pas le droit d'entrer !";
9 }
10 ?>
```

Attention, ce n'est pas la peine d'écrire :

`$autorisation_entrer == true`

car `$autorisation_entrer` est déjà une variable booléenne !

Conditions multiples

On peut utiliser les opérateurs booléens suivants :

Mot-clé	Signification	Symbole équivalent
AND	Et	&&
OR	Ou	

Exemple :

```
1 <?php
2 if ($pays == "France" OR $pays == "Belgique")
3 {
4     echo "Bienvenue sur notre site !";
5 }
6 else
7 {
8     echo "Désolés, notre service n'est pas encore disponible dans votre pays !";
9 }
10 ?>
```

La structure 'switch'

Pour éviter une succession de 'if' et 'elseif', on peut utiliser la structure 'switch' :

```
<?php
    switch ($i) {                // $i est la variable dont on veut tester la valeur
    case 0:                      // si $i vaut 0
        echo "i égal 0";        // on exécute le block correspondant
        break;
    case 1:                      // si $i vaut 1
        echo "i égal 1";        // on exécute le block correspondant
        break;
    case 2:                      // si $i vaut 2
        echo "i égal 2";        // on exécute le block correspondant
        break;
    default:                    // si $i n'est égal à aucune des valeurs précédentes
        echo "i n'est ni égal à 2, ni à 1, ni à 0.";    // on exécute ce block
    }
?>
```

Les boucles 'while' et 'for'

- Ces structures permettent de répéter un ensemble d'instructions



- Si on sait a priori combien de fois le traitement doit être répété, on utilise un 'for', sinon on utilise un 'while'
- Dans le cas du 'while', la boucle est répétée tant que la condition est vraie
- Attention à ce que la condition de boucle devienne fausse, sinon boucle infinie

Les structures 'while' et 'do .. while'

```
<?php
    while (cond) {
        ...
    }
?>
```

```
<?php
    do {
        ...
    } while (cond) ;
?>
```

```
1 <?php
2 $nombre_de_lignes = 1;
3
4 while ($nombre_de_lignes <= 100)
5 {
6     echo 'Ceci est la ligne n°' . $nombre_de_lignes . '<br />';
7     $nombre_de_lignes++;
8 }
9 ?>
```

La boucle 'for'

```
for (expr1 ; expr2 ; expr3) {  
    instructions  
}
```

où

- expr1 sert d'initialisation de la variable utilisée pour l'itération
- expr2 est la condition de continuation de l'exécution de la boucle
- expr3 est la modification de la variable à chaque tour de boucle

```
1 <?php  
2 for ($nombre_de_lignes = 1; $nombre_de_lignes <= 100; $nombre_de_lignes++)  
3 {  
4     echo 'Ceci est la ligne n°' . $nombre_de_lignes . '<br />';  
5 }  
6 ?>
```

Parcourir un tableau associatif (1)

- La boucle 'foreach' : c'est une boucle 'for' spécialisée pour les tableaux

```
foreach ($array_expression as $value) {  
    // instructions  
}
```

```
1 <?php  
2 $prenoms = array ('François', 'Michel', 'Nicole', 'Véronique', 'Benoit');  
3  
4 foreach($prenoms as $element)  
5 {  
6     echo $element . '<br />'; // affichera $prenoms[0], $prenoms[1] etc.  
7 }  
8 ?>
```

Parcourir un tableau associatif (2)

- Autre syntaxe possible pour 'foreach'

```
foreach ($array_expression as $key => $value){  
    // instructions  
}
```

```
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

Plan

- Premiers pas avec PHP
- Les variables et l'affectation
- Les tableaux
- Lecture/écriture
- Les structures de contrôle
- **Les fonctions**

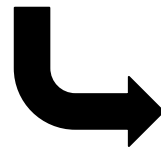
Les fonctions

- Les fonctions sont des blocs de code qui exécutent des instructions en fonction de certains paramètres
- Les fonctions ont généralement une entrée (passage par valeur) et une sortie (return ...)
- PHP propose des centaines de fonctions prêtes à l'emploi
- Si PHP ne propose pas la fonction dont on a besoin, il est possible de la créer avec le mot-clé 'function'

Fonctions prêtes à l'emploi (1)

- Traitement des chaînes de caractères

```
1 <?php
2 $phrase = 'Bonjour tout le monde ! Je suis une phrase !';
3 $longueur = strlen($phrase);
4
5
6 echo 'La phrase ci-dessous comporte ' . $longueur . ' caractères :<br />' .
  $phrase;
7 ?>
```



La phrase ci-dessous comporte 45 caractères :
Bonjour tout le monde ! Je suis une phrase !

```
1 <?php
2 $ma_variable = str_replace('b', 'p', 'bim bam boum');
3
4 echo $ma_variable;
5 ?>
```

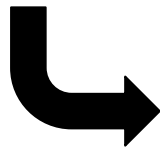


pim pam poum

Fonctions prêtes à l'emploi (2)

- Manipulation de dates

```
1 <?php
2 // Enregistrons les informations de date dans des variables
3
4 $jour = date('d');
5 $mois = date('m');
6 $annee = date('Y');
7
8 $heure = date('H');
9 $minute = date('i');
10
11 // Maintenant on peut afficher ce qu'on a recueilli
12 echo 'Bonjour ! Nous sommes le ' . $jour . '/' . $mois . '/' . $annee . 'et il est
   ' . $heure . ' h ' . $minute;
13 ?>
```



Bonjour ! Nous sommes le 02/03/2015 et il est 11 h 19.

Définition et appel de fonctions

```
1 <?php
2 function DireBonjour($nom)
3 {
4     echo 'Bonjour ' . $nom . ' !<br />';
5 }
6
7 DireBonjour('Marie');
8 DireBonjour('Patrice');
9 DireBonjour('Edouard');
10 DireBonjour('Pascale');
11 DireBonjour('François');
12 DireBonjour('Benoît');
13 DireBonjour('Père Noël');
14 ?>
```

```
1 <?php
2 // Ci-dessous, la fonction qui calcule le volume du cône
3 function VolumeCone($rayon, $hauteur)
4 {
5     $volume = $rayon * $rayon * 3.14 * $hauteur * (1/3); // calcul du volume
6     return $volume; // indique la valeur à renvoyer, ici le volume
7 }
8
9 $volume = VolumeCone(3, 1);
10 echo 'Le volume d\'un cône de rayon 3 et de hauteur 1 est de ' . $volume;
11 ?>
```

Au prochain cours

- Des révisions

Pour aller plus loin :

- <https://openclassrooms.com/courses/concevez-votre-site-web-avec-php-et-mysql>