Université Mohammed Premier
Faculté des Sciences
Département d'Informatiques
Oujda

Master Mathématiques AA et AF

Technologies du Web : PHP 5

Mme Ibtissam.ARRASSEN
2018-2019

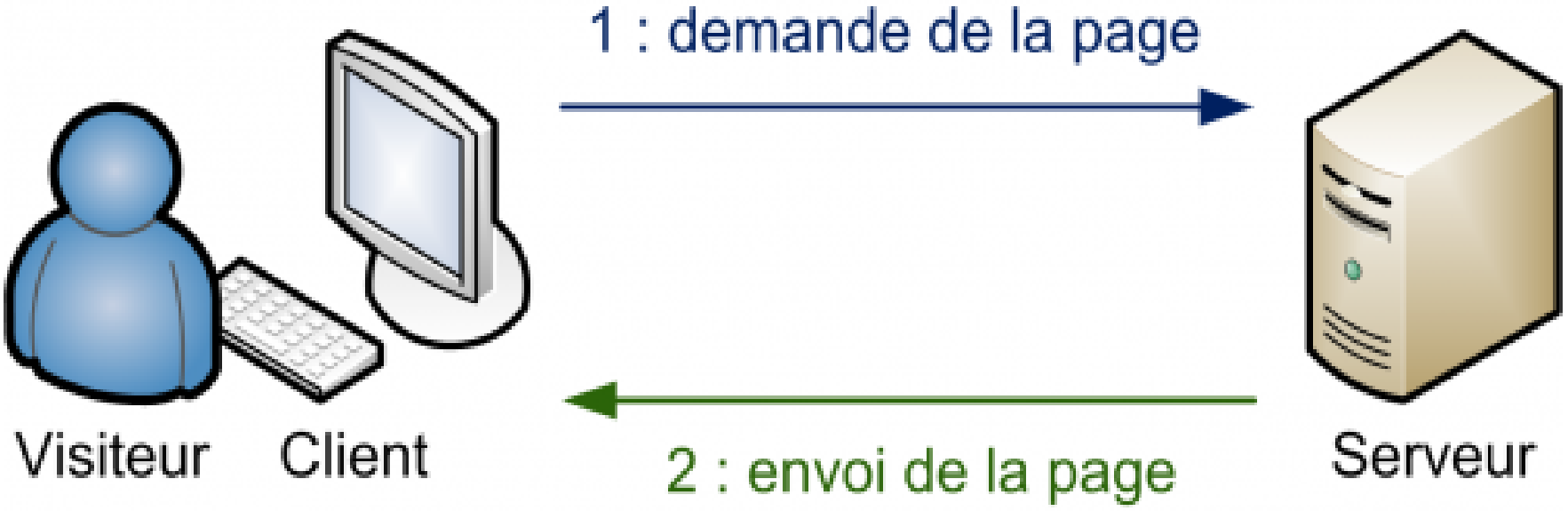
Qu'est-ce que PHP?

PHP : "Hypertext Preprocessor " .

- PHP est un langage de script open source largement utilisé.
- Les scripts PHP sont exécutés sur le serveur.
- PHP est libre pour télécharger et utiliser.

Architecture d'un site statique

- Le client demande au serveur à voir une page web.
- Le serveur lui répond en lui envoyant la page demandée.



Architecture d'un site dynamique

- Le client demande au serveur à voir une page web.
- Le serveur prépare la page spécialement pour le client.
- Le serveur lui envoie la page qu'il vient de générer.

historique

- Langage récent (crée en 1994)
- Versions utilisée :
  - 5.0 (avec une couche objet)
- Langage de script
- Langage interprété
  - Présence d'un interpréteur côté serveur
- Intégré au code HTML
- Syntaxe proche du C et du Java
- Interface simple avec beaucoup de SGBD

PHP

- Le code PHP s'insère au milieu du code XHTML.

- placer progressivement dans les pages web des morceaux de code PHP à l'intérieur du XHTML.

- Ces bouts de code PHP seront les parties dynamiques de la page, càd les parties qui peuvent changer toutes seules (c'est pour cela qu'on dit qu'elles sont dynamiques).

La forme d'une balise PHP

- Pour utiliser du PHP, on va devoir introduire une nouvelle balise... et celle-ci est un peu spéciale. Elle commence par **<?php** et se termine par **?>**.

- **<?php** /* Le code PHP se met ici */ **?>**

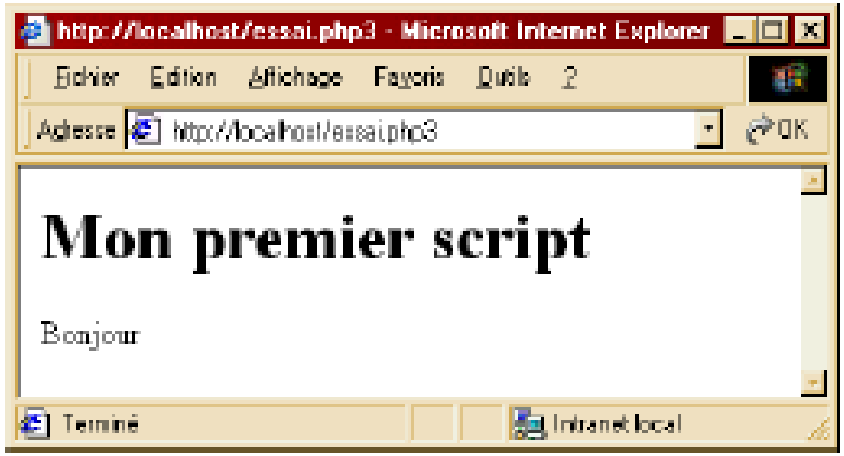
Exemple de script, code source (côté serveur) :

```
<html>
<body>
<h1>Mon premier script</h1>
<?php echo "Bonjour\n"; ?>
</body>
</html>
```

Autre écriture du même script :

```
<?php
echo "<html>\n<body>\n";
echo "<h1>Mon premier script</h1>\n";
echo "Bonjour\n";
echo "</body>\n</html>\n";
?>
```

Résultat affiché par le navigateur :



Variables PHP

- Une variable peut avoir un nom abrégé (comme x et y) ou un nom plus descriptif (age, carname, total_volume).

Règles pour les variables PHP:

- Une variable commence par le signe $, suivi du nom de la variable
- Un nom de variable doit commencer par une lettre ou le caractère de soulignement
- Un nom de variable ne peut pas commencer par un nombre
- Un nom de variable ne peut contenir que des caractères alphanumériques et des caractères de soulignement (A-z, 0-9 et _)
- Les noms de variable sont sensibles à la casse ($age et $AGE sont deux variables différentes)

Variables, types et opérateurs

- En PHP, une variable commence par le signe $, suivi du nom de la variable
- Les variables sont de type:

- Entier (integer)
- Réel (double)
- Chaîne de caractères (string)
- Tableau (array)
- Booléen (boolean)
- Objet (object) : Un objet est un type de données qui stocke des données et des informations sur la façon de traiter ces données. En PHP, un objet doit être explicitement déclaré.

Variables, types et opérateurs

- Il est possible de convertir une variable en un type primitif grâce au **cast().**

  Exemple:

  - $str = ’12’;
  - $nbr = (int) $str;

- Quelques fonctions:

  - **Empty ($var**) : renvoie vrai si la variable est vide
  - **Isset ($var**) : renvoie vrai si la variable existe
  - **Unset ($var**) : détruit une variable

Variables, types et opérateurs

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

Syntaxe : ${$var}=valeur;

Exemple :

- $toto = 'foobar' ;
- ${$toto} = 2002 ;
- Echo $foobar; //la variable $foobar vaut 2002

- En php, il existe la notion de variable globale et variable locale.

Constantes

- la valeur est fixée une fois pour toute et n'est pas modifiable
- Les constantes ne portent pas le symbole $ en début d'identificateur.

- **Syntaxe:**
  - **Define (variable , valeur) ;**

- **Exemple:**
  - **Define (MY_YEAR , 1980);**
  - **Echo MY_YEAR ; //affiche 1980**

Les conditions

- **Les symboles de comparaisons :**

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

Les conditions

- La structure If... Else
- La structure switch … case ….

Les boucles

- **Les boucles:**
  - While
  - For

Afficher du texte

- Les fonctions d'affichage:

  - **Echo (" Bonjour $name ");**

    //écriture dans le navigateur

  - **Print (" Bonjour $name ");**

  - **Printf (" Bonjour %s " , $name);**

    //écriture formatée comme en C

Affichage

- Les "“ permettent l’évaluation des variables et caractères spéciaux
- les ‘ ’ ne le permettent pas.

- **Exemple**:

```
$nom=   " Saad";
Echo  "Nom: $nom“  ; //Affiche      Nom: Saad
Echo  ‘Nom: $nom’  ; //Affiche      Nom: $nom
```

- Concaténation se fait par un ‘ . ‘
- **Exemple**:

```
<?php
$foo = "Hello";
$bar = "Word";
echo    $foo.$bar;        //concaténation par un point
?>
```

Les commentaires

- Il existe 2 types de commentaires :
  - Les commentaires monolignes:

```
<?php
echo "J'habite au Maroc.";// Cette ligne indique où j'habite
// La ligne suivante indique mon âge
echo "J'ai 92 ans.";
?>
```

  - Les commentaires multilignes:

```
<?php
/* La ligne suivante indique mon âge
Et ma ville natale*/
echo "J'ai 92 ans.";
Echo « Je suis né à Oujda »;
?>
```

Tableaux

- Une variable tableau est de type array.
- Un tableau en php accepte des éléments de tout type.
- Les éléments d'un tableau peuvent être de types différents et sont séparés d'une virgule.
- Un tableau peut être intialisé avec la syntaxe array:

  $tab = array ('foobar‘ , 2002 , 20.5 , $name) ;

- Mais, il peut aussi être initialisé au fur et à mesure.

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

```
<?php
$prenoms[] = 'François';
$prenoms[] = 'Michel';
$prenoms[] = 'Nicole';
?>
```

Parcours d'un tableau

$tab=array ('Hugo' , 'Jean' , 'Mario')

- Exemple 1:

```
$i=0;
While ( $i < count  ($tab)  ) {  //count() retourne le nombre d'éléments
    Echo  $tab[$i].'\n' ;
    $i++;
}
```

- Exemple 2:

```
Foreach ( $tab  as  $elem ) {
    Echo  $elem.'\n' ;
}
```

La variable **$elem** prend pour valeurs successives tous les éléments du **$tab**.

Quelques fonctions

- **Count ($tab), Sizeof** : retournent le nombre d'éléments du tableau.
- **List ($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.

Tableaux associatifs

- Les tableaux associatifs sont des tableaux qui utilisent des clés nommées que vous leur assignez.

- Pour les créer, on utilisera la fonction array, on va mettre "l'étiquette" devant chaque information :

```
<?php
   // On crée notre   array   $age

   $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
?>
```

Tableaux associatifs

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

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

- Afficher un tableau associatif

```
<?php
$age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo "Peter is " . $age['Peter'] . " years old.";
?>
```

Rechercher dans un tableau

- Nous allons voir trois types de recherches, basées sur des fonctions PHP :

  - **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

array_key_exists

- **Vérifier si une clé existe dans l'array :**

```
<?php
$coordonnees = array (
        'prenom' => 'François',
        'nom' => 'Dupont',
        'adresse' => '3 Rue du Paradis',
        'ville' => 'Marseille');

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

in_array

- **Vérifier si une valeur existe dans l'array :**

```
<?php
   $fruits = array ('Banane', 'Pomme', 'Poire', 'Cerise', 'Fraise', 'Framboise');

        if (in_array('Myrtille', $fruits))
        {
                echo 'La valeur "Myrtille" se trouve dans les fruits !';
        }

        if (in_array('Cerise', $fruits))
        {
                echo 'La valeur "Cerise" se trouve dans les fruits !';
        }
?>
```

array_search

- **Récupérer la clé d'une valeur dans l'array :**

```
<?php
$fruits = array ('Banane', 'Pomme', 'Poire', 'Cerise', 'Fraise', 'Framboise');

    $position = array_search  ( 'Fraise‘ ,  $fruits );
    echo '"Fraise" se trouve en position ' . $position . '<br/>';

    $position = array_search  ( 'Banane‘ ,  $fruits );
    echo '"Banane" se trouve en position ' . $position;
?>
```

Les fonctions en PHP

Fonctions PHP définies par l'utilisateur

- ✓ Outre les fonctions PHP intégrées, nous pouvons créer nos propres fonctions.
- ✓ Une fonction est un bloc d'instructions qui peuvent être utilisées à plusieurs reprises dans un programme.
- ✓ Une fonction ne s'exécute pas immédiatement lorsqu'une page est chargée.
- ✓ Une fonction sera exécutée par un appel à la fonction.

Fonctions

Remarque:

- un nom de fonction peut commencer par une lettre ou un trait de soulignement (pas un chiffre).

Astuce:

- Donnez à la fonction un nom qui reflète ce que fait la fonction!

Arguments de la fonction PHP

- Les informations peuvent être transmises à des fonctions via des arguments. Un argument est comme une variable.

- On peut ajouter autant d'arguments qu'on souhaite, il suffit de les séparer par une virgule.

```
<?php
function     familyName  ($fname) {
        echo "$fname   Rousseau<br>";
       }

familyName ("Jani");
familyName ("Hege");
familyName ("Stale");
familyName ("Kai Jim");
familyName ("Borge");

?>
```

Valeur d'argument par défaut

- L'exemple suivant montre comment utiliser un paramètre par défaut.
- Si nous appelons la fonction **setHeight ()** sans arguments, elle prend la valeur par défaut comme argument:

```
<?php
function setHeight   ($minheight = 50) {

        echo "The height is : $minheight <br>";
}

setHeight (350);
setHeight ();  // utilise la valeur par défaut 50
setHeight (135);
setHeight (80);
?>
```

Fonctions PHP - Retour des valeurs

- Pour laisser une fonction renvoyer une valeur, utilisez l'instruction **return**:

- ```
  <?php
  function sum($x, $y) {
      $z = $x + $y;
      return $z;
  }

  echo  "5 + 10 = " . sum(5, 10) . "<br>";
  echo  "7 + 13 = " . sum(7, 13) . "<br>";
  echo  "2 + 4 = " .  sum(2, 4);
  ?>
  ```
  ```