

Développement Web avec PHP

Notre premier exemple PHP

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <title>Page PHP</title>
7 </head>
8
9 <body>
10
11     <h1>
12         <?php
13         echo "Hello world!";
14         ?>
15     </h1>
16
17 </body>
18
19 </html>
```

Les commentaires

```
12 <?php
13 echo "Hello world!";
14
15 // un commentaire
16
17 /*
18 un commentaire php sur plusieurs lignes
19 */
20
21 ?>
```

La fonction « echo »

- Elle permet d'afficher un texte.

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Page PHP</title>
7  </head>
8
9  <body>
10     <?php
11         echo "<h1>Hello world!</h1>";
12
13     ?>
14 </body>
15
16 </html>
```

Les variables

- Une variable est un élément du programme qui peut prendre plusieurs valeurs au cours de l'exécution du programme.
- Absence de typage des variables

Les variables

➤ Les conventions de nommage:

- Le nom de la variable est précédé du signe \$.
- Sont obligatoirement des chaînes de caractères non numérique.
- Ne doivent pas être accentués.
- Ne contiennent pas un espace.
- Attention, PHP est sensible à la casse.

Les variables

```
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5      <meta charset="UTF-8">
6      <title>Variables</title>
7  </head>
8
9  <body>
10     <?php
11         //fonction prédéfinie qui fournit la date système
12         $maDate = date("d-m-y");
13         echo "Hello world! </br>";
14         // concaténation
15         echo 'on est le ' . $maDate;
16
17     ?>
```

Les constantes

➤ Une constante est une référence à une valeur qui ne change pas.

➤ Exemple:

```
16 // titre du site  
17 define('site', 'www.emsi.ma');
```

➤ Les constantes sont ensuite appelées de cette façon:

```
18 echo site;
```


Les chaînes de caractères

```
10 <?php
11 //fonction prédéfinie qui fournit la date système
12 $maDate = date("d-m-y");
13 // concaténation
14 echo "on est le $maDate";
15 echo 'on est le $maDate';
16 echo 'on est le ' . $maDate;
```

➤ Résultat:

- Les deux lignes de code 14 et 16 afficheront le contenu de la variable.
- la ligne 15 \$maDate n'est pas interprétée comme une variable

Les Opérateurs numérique

| Opérateur | Signification | Exemple |
|-----------|----------------|----------------------------|
| + | addition | <code>\$op1 + \$op2</code> |
| - | soustraction | <code>\$op1 - \$op2</code> |
| * | multiplication | <code>\$op1 * \$op2</code> |
| / | division | <code>\$op1 / \$op2</code> |
| % | modulo | <code>\$op1 % \$op2</code> |

Les opérateurs de comparaison

| Opérateur | Signification | Exemple |
|-----------|--|--------------------------------|
| == | égalité (attention ! il faut deux "égals") | <code>\$op1 == \$op2</code> |
| === | identique (égal et même type) | <code>\$op1 === \$op2</code> |
| < | strictement inférieur | <code>\$op1 < \$op2</code> |
| > | strictement supérieur | <code>\$op1 > \$op2</code> |
| <= | inférieur ou égal | <code>\$op1 <= \$op2</code> |
| >= | supérieur ou égal | <code>\$op1 >= \$op2</code> |
| != ou <> | différent | <code>\$op1 != \$op2</code> |

Les opérateurs logiques

| Opérateurs | ET | OU inclusif | OU exclusif | NON |
|------------|---------------|--------------|-------------|-----|
| Syntaxe | "&&" ou "and" | " " ou "or" | xor | ! |

Les structures de contrôle - If / elseif / else

➤ Avec plusieurs possibilités:

```
21
22 ✓  if (condition1) {
23      # code...
24 ✓  } elseif (condition2) {
25      # code...
26 ✓  } else {
27      # code...
28  }
29
```

Les structures de contrôle - switch

```
22  switch ($variable) {  
23      case 'value1':  
24          # code...  
25          break;  
26      case 'value2':  
27          # code...  
28          break;  
29      default:  
30          # code...  
31          break;  
32  }  
33
```

Les structures de contrôle - switch

➤ Exercice:

- la fonction `date()` ne nous retourne pas les mois sous forme littérale en français, créez un programme qui permet d'afficher le mois en cours en français.

```
$mois_en_cours = date("m");
```

Les boucles - while

- Syntaxe:

```
<?php
    while(condition)
    {
        instruction 1;
        instruction 2;
        ...
    }
?>
```


Les boucles - FOR

- Syntaxe:

```
22     for ($i=0; $i <=5 ; $i++) {  
23         # code...  
24     }
```

Tableaux-Définition

- Un tableau est une liste d'éléments organisée en couples : clé + valeur.
 - Clé: entier (tableau numérique - la clé est un indice) ou chaîne (tableau associatif).
 - La valeur associée à la clé est de type indifférent.
- Il est possible de stocker des éléments de types différents dans un même tableau.

les tableaux à indice numérique

➤ Exemple1:

```
<?php
    $mois = array("janvier","février","mars",
    "avril");
?>
```

➤ Exemple2:

```
<?php
    $mois = array();
    $mois[0] = 'janvier ';
    $mois[1] = 'février ';
    $mois[2] = 'mars';
    $mois[3] = 'avril';
?>
```

Tableaux associatifs

```
<?php
    $article = array(
        'Numero' => 77,
        'Nom' => 'clavier',
        'Prix' => 7);
?>
```

➤ ou:

Parcourir un tableau avec foreach

- Syntaxe:

```
foreach ($tableau as $cle => $valeur) {  
    instructions à exécuter;  
}
```

les tableaux multidimensionnels

➤ Exemple

```
<?php
    $personnes = array(
        1 => array('prenom' => 'mohamed', 'nom' =>
            'naji', 'telephone' => '0669663315'),
        2 => array('prenom' => 'youssef', 'nom' =>
            'bekri', 'telephone' => '0661253348')
    );

    echo $personnes[2]['prenom']; //youssef
?>
```

Les fonctions

- Une fonction est une série d'instructions qui effectue des actions et qui retourne une valeur.
- Une fonction ne s'exécutera pas automatiquement lors du chargement d'une page.
- Une fonction sera exécutée par un appel à la fonction

```
<?php
    function mafonction(param1,...,paramN)
    {
        Instructions
        [return valeur retournee;]
    }
?>
```

Les fonctions

➤ Exemple (fonction sans paramètres):

```
<?php
function afficherMessage(){
    echo "Bonjour tout le monde";
}
afficherMessage();
?>
```

➤ Exemple (avec paramètres):

```
<?php
function calculerSomme($a,$b){
    $somme=$a+$b;
    echo "La somme des deux nombres est: $somme";
}
calculerSomme(10,30);
?>
```



Les fonctions – passage par valeur et par référence

```
<?php
function ajouterSix($nb){
    $nb+=6;
}
function ajoutercinq(&$nb){
    $nb+=5;
}

$nbOrigine=10;
ajouterSix($nbOrigine);
echo $nbOrigine;////affichera 10

ajoutercinq($nbOrigine);
echo $nbOrigine;//affichera 15
?>
```

Les fonctions – qui retourne une valeur

```
<?php

function calculerSomme($a,$b){
    $somme=$a+$b;
    return $somme;
}
$valRetour=calculerSomme(10,30);
echo "La somme des deux nombres est: $valRetour";

?>
```

Les fonctions – Définir une valeur par défaut :

```
<?php

function definirHauteur($hauteur = 50) {
    echo "La hauteur est: $hauteur <br>";
}

definirHauteur(30);
definirHauteur(); // utilisera la valeur par défaut de
50
definirHauteur(15);
?>
```

Forms

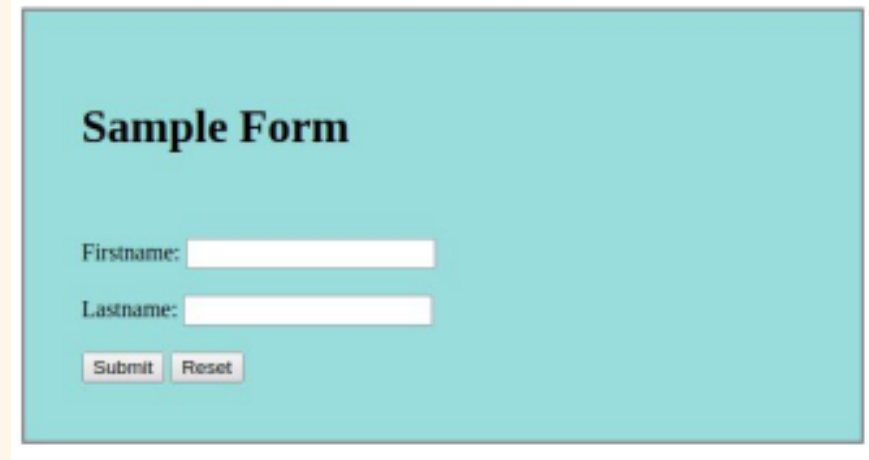
- Les formulaires sont des composants spéciaux qui permettent aux visiteurs de votre site de fournir divers informations sur la page HTML.
- Les formulaires incluent généralement certains éléments de saisie pour recueillir des informations.
- Cette fois, nous allons parler du traitement des formulaires côté serveur en utilisant PHP.



Forms

index.html

```
<form name="myForm" id="myForm">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn">
  <input type="reset" name="resetBtn">
</form>
```



Sample Form

Firstname:

Lastname:

Forms

- **Action** : c'est un attribut de la balise form qui définit le chemin d'envoi des données de la formulaire.
- **Method**: c'est un attribut qui définit comment ces les données de la formulaire seront communiqué avec la partie serveur on peut identifier deux valeur POST et GET.

Forms

index.html

```
<form name="myForm" id="myForm" ?>
action="process.php" method="GET">
  <br><br>
  <span id="firstname">Firstname: </span>
  <input type="text" name="firstname"><br><br>
  <span id="lastname">Lastname: </span>
  <input type="text" name="lastname"><br><br>
  <input type="submit" name="submitBtn">
  <input type="reset" name="resetBtn">
</form>
```

process.php

```
<?php
function welcome($fname, $lname)
{
    echo '<h1>Welcome ' . $fname . ' ' . $lname . '!</h1>';
}

welcome($_GET['firstname'], $_GET['lastname'])
```

Sample Form

Firstname:

Lastname:

Comment traiter les données de la formulaire

```
$_GET['firstname']
```



This array element holds the value for firstname field in the form

```
$_GET['lastname']
```



This array element holds the value for lastname field in the form

PHP et HTML dans le même fichier

index2.php

```
<?php
if ($_GET['firstname'] != null)
{
    function welcome($fname, $lname)
    {
        echo '<h1>Welcome ' . $fname . ' ' . $lname . '!</h1>';
    }

    welcome($_GET['firstname'], $_GET['lastname']);
}
else {
?>
<!doctype html>
<html>
<head>
<title>Form Validation</title>
```

index2.php continued

```
<link rel="stylesheet" href="css/style.css">
</head>
<body>
<div id="container">
<h1>Sample Form</h1>
<form name="myForm" id="myForm" action="index2.php" method="get">
    <br><br>
    <span id="firstname">Firstname: </span>
    <input type="text" name="firstname"><br><br>
    <span id="lastname">Lastname: </span>
    <input type="text" name="lastname"><br><br>
    <input type="submit" name="submitBtn"><input type="reset" name="resetBtn">
</form>
</div>
</body>
</html>
<?php
}
?>
```

