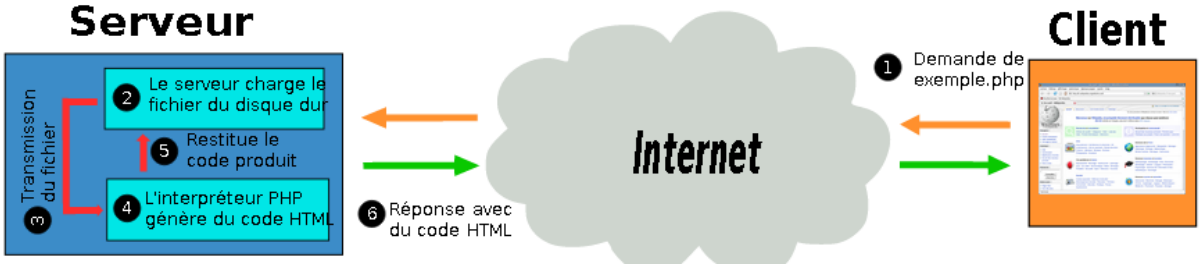
Travaux Pratiques - M2105 : Web dynamique

Introduction à PHP

Le but de ce TP est de vous familiariser avec le langage PHP, utilisé pour la génération de pages Web dynamiques sur un serveur web.

Dans une utilisation Web, l'exécution du code PHP se fait ainsi : lorsqu'un visiteur demande à consulter une page Web, son navigateur envoie une requête à un serveur HTTP. Si la page contient du code PHP, l'interprèteur PHP du serveur le traite et renvoie du code XHTML généré.

Voici un schéma qui illustre le principe :



L'utilisation la plus répandue est celle qui fait de PHP un générateur de page Web (contenant du code HTML, CSS, JavaScript, etc.). Ce langage permet donc de générer des pages au contenu dynamique (différent en fonction des données passées en paramètres, souvent par le biais de formulaires).

1) Installation serveur Web Apache et PHP

Vérifiez que le serveur Web Apache2 est bien installé ainsi que les modules en charge du php.

Pour cela tapez les commandes :

aptitude install apache2 apache2-utils php5 php5-dev php5-gd

puis lancez le serveur Web : /etc/init.d/apache2 restart

Le répertoire où se trouvent les fichiers du serveur web est : /var/www.

Créez le fichier Test.html suivant dans /var/www/:

```
<!DOCTYPE html>
<html> <body>
Hello !
</body> </html>
```

Ouvrez la page Test.html en passant par votre serveur web lancé sur l'adresse IP du localhost : 127.0.0.1

http://127.0.0.1/Test.html

2) Structure d'un fichier PHP

Toute commande PHP se termine par un **point-virgule**!

Tout code PHP doit être inclus dans une balise **<?php ... ?>**

Editez un fichier Hello.php contenant la ligne suivante :

```
<?php echo "Ici commence notre apprentissage de PHP !" ;
?>
```

Testez votre script php à l'aide de la commande php :

php Hello.php

3) Interaction avec XHTML

Il y a deux façon de procéder pour mélanger du PHP et du (X)HTML :

- Soit insérer du PHP dans du HTML

```
<html>
  <body>
   <p> Bonjour</p>
   <?php echo "Tiens, du PHP ..."; ?>
   <p> Au revoir</p>
  </body>
</html>
```

Créez une page html HelloPhp1.php contenant le code ci-dessus.

Placer la page dans l'arborescences du site web Apache : dans le répertoire /var/www

Lancez un navigateur et ouvrez la page 127.0.0.1/HelloPhp1.php . Commentez.

- Soit insérer du HTML dans du PHP

```
<?php
  echo "<html>\n";
  echo "  <body>\n";
  echo "    <p> Je fais du HTML avec du PHP !</p>\n";
```

```
  echo "  </body>\n";

  echo "</html>\n";

?>
```

Créez une page html HelloPhp2.php contenant le code ci-dessus.

Placer la page dans l'arborescences du site web Apache : dans le répertoire /var/www

Lancez un navigateur et ouvrez la page127.0.0.1/HelloPhp2.php . Commentez.

4) Insérer des commentaires en PHP

Pour insérer des commentaires dans votre code, vous devez suivre une des syntaxes suivantes :

/* Votre commentaire sur plusieurs lignes .... */
// Votre commentaire
#Votre commentaire

```
      <?php
/* Je fais du HTML avec du PHP */

  echo "<html>\n";
  echo "  <body>\n";
  echo "    <p> Je fais du HTML avec du PHP !</p>\n";
  echo "  </body>\n";
  echo "</html>\n";

?>
```

5) Les variables

Une variable commence par un dollar : **$**.
Les noms de variables sont **sensibles à la casse** !
$variable et **$Variable** sont deux variables différentes.

Les chaînes de caractères doivent être insérées dans des guillemets.

Exemple : *Testez l'exemple suivant et vérifier que la variable $prenom affiche bien Jean*

```
<?php
$prenom = "Jean" ;

echo "<html>\n";
echo "  <body>\n";
echo "<p> Je m'appelle" .$prenom. "</p>\n";
echo "  </body>\n";
echo "</html>\n";

?>
```

6) Les caractères spéciaux

Par contre, si vous insérez du HTML dans du PHP, vous devez traiter à part tous les caractères spéciaux si vous voulez que PHP ne les interprète pas. C'est par exemple le cas des guillemets dans cet exemple :

Il faut protéger les guillements par un \ avant les guillemets

Créez une page web TestCar.hml contenant ce code html. Testez cette page dans un navigateur

```
<!DOCTYPE html>
<html>
<body>

<table border="1" style="width:300px">
<tr>
  <td>Jill</td>
  <td>Smith</td>
  <td>50</td>
  </tr>
<tr>
  <td>Eve</td>
  <td>Jackson</td>
  <td>94</td>
</tr>
<tr>
  <td>John</td>
  <td>Doe</td>
  <td>80</td>
</tr>
</table>

</body>
</html>
```

Créez un fichier TestCar.php pour que le tableau soit généré en php en protégeant les guillemets dans le code php :

```
<!DOCTYPE html>
<html>
<body>
```

```
<?php
echo "texte du tableau…";
?>

</body>
<html>
```

Ouvrez lez fichier TestCar.php et vérifiez que le tableau s'affiche correctement.

7) La concaténation

L'opérateur utilisé pour la concaténation de chaînes de caractères est le point.

```
<?php

$nom = "Pierre" ;
$prenom = "Jean" ;
echo "Mon nom est" . $prenom . " " . $nom ;

?>
```

Ce qui aura comme résultat : *Mon nom est Jean Pierre*.

Testez cet exemple.

8) Les tableaux

On distingue deux types de tableaux en PHP : les **tableaux indicés** et les **tableaux associatifs**.

- Tableaux indicés

Un tableau indicé est un tableau où chaque élément correspond à un indice en fonction de sa position dans le tableau.

Par exemple, si je crée un tableau contenant des fruits :

```
<?php

$fruits = array ('banane', 'cerise', 'poire', 'pomme') ;

?>
```

Pour avoir accès aux éléments, je dois faire appel à eux par leur indice. Il faut savoir qu'un tableau indicé commence toujours à 0 !

Testez le code suivant dans un fichier TestTableau.php

```
<?php

      $fruits = array ('banane', 'cerise', 'poire', 'pomme') ;
   echo  $fruits[0] ;
   echo $fruits[3];
?>
```

Gestion de la date

Ajouter la date (fonction prédéfinie)

- $date = **date("d/m/y")** ; // Date du jour
- $heure = **date("H:i")** ; // et l'heure

Réalisez un fichier php qui affiche :

```
Nous sommes le :<affiche la date>

Il est : <affiche l'heure>

Salut !
```

9) Gestion des formulaires : méthodes GET/POST

On souhaite écrire un formulaire qui demande le nom et l'âge de l'utilisateur. Le bouton submit de ce formulaire provoquera l'affichage d'une page qui saluera l'utilisateur avec cette phrase : « Bonjour machin, vous avez xx ans... » (avec les bonnes valeurs, bien entendu).

Dans un fichier TestFormulaire.html testez le code suivant permettant de réaliser le formulaire et expliquez le principe de la méthode get et le nom des variables transmises au serveur http.

```
<html><body>
    <form method="get" action=" TestFormulaire.php">
    <table>
    <tr>
      <td align="right">Votre nom</td>
      <td><input name="nom" /></td>
    </tr><tr>
      <td align="right">Votre age</td>
      <td><input name="age" /></td>
    </tr><tr>
      <td align="center" colspan="2"><input type="submit" value="Envoyer" />
    </tr>
    </table>
    </form>
    </body></html>
```

Créez un fichier TestFormulaire.php contenant le code suivant :

```
<html>
<body>
        <?php
        echo "Bonjour ".$_GET["nom"]. " vous avez ".$_GET["age"]." ans.\n";
        ?>
</body>
</html>
```

Expliquez le fonctionnement du script TestFormulaire.php.
A quoi correspondent les variables $_GET["nom"] et $_GET["age"] ?

Expliquez comment les données du formulaire sont transmises avec la méthode GET après un click sur le bouton « Envoyer ».

Remplacer dans les fichiers TestFormulaire.html et TestFormulaire.php la méthode GET par la méthode POST (changer get en post dans l'ensemble des fichiers y compris pour les variables en respectant la casse).

Quelle différence observez-vous lors du click sur le bouton « Envoyer » ? Expliquez.

10) Structures de contrôle : boucles et tests

La boucle foreach

Les boucles permettent d'effectuer des opérations d'itération. On distingue essentiellement les commandes **while** et **foreach**.
La boucle **foreach** va vous permettre de parcourir un tableau.

Tester l'exemple:

```
        <?php
$fruits = array ('banane', 'cerise', 'poire', 'pomme') ;

  foreach ($fruits as $un_fruit)
   {
     echo "Fruit: ".$un_fruit."<br/>" ;
   }
 ?>
```

Les tests

La structure d'un test est la suivante :

```
        if (condition) {

  instruction à effectuer si la condition est vraie

 }
```

```
else {

  instruction à effectuer si la condition précédente n'a pas été vérifiée

}
```

Le **else** est optionnel, et vous pouvez imbriquer des **if** à l'intérieur d'autres **if**.
Il est aussi possible d'avoir plus de conditions, que vous pouvez exprimer avec **elseif**.

Testez le scipt suivant.

```
<?php

    $total = 25 ;
    $chiffre=30 ;

    if ($chiffre < $total) {
       echo "Trop petit !" ;
    }
    elseif ($chiffre > $total) {
       echo "Trop grand !" ;
    }
   else {
      echo "Gagné !" ;
    }

?>
```

Adaptez ce script pour pouvoir saisir le nombre $chiffre dans un formulaire.