Initiation au PHP

Serveur et formulaires

Quelques variables serveur utiles

Code Source

```
<html>
  <head>
    <title>Quelques variables serveur utiles</title>
  </head>
  <body>
    <h1>Quelques variables serveur utiles</h1>
    <br /><br />
    Par rapport à la racine du site, la page actuelle se situe dans :✂
    <?php echo $_SERVER['PHP_SELF']; ?><br />
    <br />
    Vous venez de la page : <?php echo $_SERVER['HTTP_REFERER']; ?><br />
    <br />
    Votre adresse IP est : <?php echo $_SERVER['REMOTE_ADDR']; ?><br />
    <br />
    Votre navigateur est identifié par : ✂
    <?php echo $_SERVER['HTTP_USER_AGENT']; ?><br />
    <br />
    La langue de votre navigateur est : ✂
    <?php echo $_SERVER['HTTP_ACCEPT_LANGUAGE']; ?>
  </body>
</html>
```

Récupérer les données d'un formulaire

Généralités

Pour récupérer les données en provenance d'un formulaire html, PHP met à votre disposition quelques fonctions bien utiles.

Dans la création d'un formulaire HTML vous avez deux façons de l'envoyer vers un script PHP, soit par **GET** ou **POST**. Vous indiquez ce choix avec l'attribut `method` de la balise `form`. L'attribut `action` vous permet de spécifier quel script sera déclenché sur le serveur.

```
<form action="script.php" method="GET">

</form>
```

Récupération des données

Si les données HTML sont transmises avec la méthode **`GET`**, dans votre script PHP vous utilisez: **`$_GET`** ou **`$_REQUEST`**

Si les données HTML sont transmises avec la méthode **`POST`**, dans votre script PHP vous utilisez: **`$_POST`** ou **`$_REQUEST`**

Vous voyez que dans les deux cas, **`$_REQUEST`** peut être utilisé.

Exemple :

un champ nommé prenom:

```
<input type="text" name="prenom" />
```

Si il est transmis en `GET` sera récupéré de la manière suivante `$_GET["prenom"]` ou `$_REQUEST["prenom"]`

Si il est transmis en `POST` sera récupéré de la manière suivante `$_POST["prenom"]` ou `$_REQUEST["prenom"]`

On peut tester si l'envoi est fait par POST ou GET :

```
if ( $_SERVER['REQUEST_METHOD'] == "GET") {


}
```

Gestion des données sur le serveur

DANS UN FORMULAIRE SIMPLE

Voilà un exemple et son code

```
<form action="script.php" method="post">
  Nom : <input type="text" name="nom" size="20" maxlength="40" /> <br />
  Prenom : <input type="text" name="prenom" size="20" maxlength=40 />
  <input type="submit" value="ENVOYER" />
</form>
```

Dans le fichier "**script.php**", nous récupérons directement les variables issues du nom des champs du formulaire.

```
/* Récupère le nom et le prénom automatiquement */

echo "bonjour" . $_REQUEST['prenom'] . " " . $_REQUEST['nom'] . "!<br />";
```

Le résultat est alors : bonjour Denis Le Roy !

Dans ce cas, vous pouvez lire les données du formulaire en utilisant les tableaux $_REQUEST ou $_POST.

Transmettre des champs de formulaire sous la forme d'un tableau

Voilà un exemple et son code :

```
<form action ="check.php" method="post">
  Tu utilises internet plutôt le :<br />
  <input type="checkbox" name="Fjour[]" value="Lundi" />Lundi<br />
  <input type="checkbox" name="Fjour[]" value="Mardi" />Mardi<br />
  <input type="checkbox" name="Fjour[]" value="Mercredi" />Mercredi<br />
  <input type="checkbox" name="Fjour[]" value="Jeudi" />Jeudi<br />
  <input type="checkbox" name="Fjour[]" value="Vendredi" />Vendredi<br />
  <input type="submit" name="Envoyer" value="ENVOYER" />
</form>
```

Vous remarquerez que nous passons en fait un tableau **Fjour[]** pour rassembler l'ensemble des valeurs possibles des différentes cases à cocher.

Dans FORM ACTION nous postons cela au fichier **check.php** qui est composé comme cela :

```php
<?php
   echo "Tu surfes sur le web en semaine plutôt le : ";
   foreach ($_REQUEST["Fjour"]) as $jour)      /* Lecture du tableau */
   {
        echo " $Jour - ";
   }
?>
```

L'UTILISATION DES CHAMPS CACHES

```html
<form action="http://www.serveur.com/page.php" method="post">
  Votre e-mail : <input type="text" name="email" />
  <input type="hidden" name="secret" value="texte à passer discrètement" />
  <input type="submit" value="OK" />
</form>
```

Après avoir rempli la case "Votre Email" du formulaire et en cliquant sur "OK", celui-ci appellera la page "**page.php**" qui recevra les variables "Email" et "secret".

L'utilisation du type "**hidden**" permet de passer des variables discrètement, sans qu'elles soient affichées à l'écran, ***mais elles sont visibles dans le code.***

Passer des variables par l'URL (Method GET)

Pour passer une variable d'une page à une autre, il faudra ajouter un **point d'interrogation** (**?**) à l'URL suivi de la variable à transmettre. Pour passer plusieurs variables, il suffira de les séparer par le signe **&** :

```
http://www.monserveur.com/page.php?prenom=bill&lang=fr
```

Cela correspond à l'utilisation de la méthode **GET**

LA PAGE FORMULAIRE ET LE SCRIPT PEUVENT NE FAIRE QU'UN:

```php
<form action="<?php print $_SERVER['PHP_SELF'] ?>"method="post">
  <input type="text" name="zoneDeTexte" size="40" maxlength="40" />
  <input type="submit" name="btnEnvoyer" value="envoyer" />
</form>
```

La variable **$_SERVER['PHP_SELF']** est créée par le système et contient le nom du script courant.

Notre page affichée est donc construite par notre script PHP et à l'envoi du formulaire, c'est le même script qui est executé, pas nécessairement la même page qui est affiché.

Remarque : La variable **$_SERVER['PHP_SELF']** pose parfois des problèmes chez certains hébergeurs, elle contient l'URL complète avec le nom de domaine. Pour travailler en local, pas de problèmes.

Pour savoir ce qu'elle contient il suffit de faire un **'echo $_SERVER['PHP_SELF']** dans un script PHP.

Quel est l'intérêt d'utiliser le même script pour envoyer et recevoir ? Vous allez vite comprendre. Imaginons le formulaire qui lit ses informations dans une base de données, cela permet à l'utilisateur de corriger, supprimer et créer des fiches.

On insère des boutons "création", "modification"... dans le formulaire :

```
<input type="submit" name="delete" value="Supprimer" />
<input type="submit" name="create" value="Créer" />
<input type="submit" name="update" value="Modifier" />
```

Le script gère le bouton qui a été utilisé pour transmettre le formulaire afin de mettre en œuvre les actions adéquates. On utilise ici la fonction **isset** qui retourne **true** si la variable a reçu une valeur.

```
if ( isset( $_POST['update'])) { // mise à jour
      echo "mise a jour";
} elseif ( isset($_POST['delete']))) {
      echo "suppression";
} elseif ( isset($_POST['create']))) {
      echo "Création";
}
```

Ensuite dans le corps de la page on affiche la fiche modifiée ou une fiche vierge avec des valeurs par défaut par exemple. Dans le cas de la suppression, il faut faire des choix soit la précédente ou la suivante, en testant que ce n'est pas suivant le cas la première ou la dernière.

Le résultat devient alors :

Exercice

Reprenez le formulaire que vous avez realisé dans la séance précédente (JavaScript).

Dans ce formulaire , vous devez modifier l'attribut action de la balise form pour indiquer l'adresse d'un script PHP.

```
<form action="monscript.php">
```

Puis créer ce script PHP permettant d'afficher l'ensemble des valeurs transmises.

```
$_REQUEST["nom_du_input"]
```

Cette instruction permet de récupérer la valeur du parametre nom_du_input