Traiter une soumission de formulaire en PHP

En PHP, la manière principale de communiquer des données au serveur est pas la soumission de formulaire. Voyons un exemple simple avec un formulaire de connexion.

Le formulaire

Commençons déjà par écrire notre formulaire dans un document HTML : `login.html`

Base du formulaire

```
<form method="post" action="login.php">


</form>
```

Nous allons utiliser la méthode HTTP POST car nous devons envoyer des données au serveur. Ensuite dans l'attribut `action` nous mettons le chemin sur le serveur denotre script PHP qui va gérer la soumission du formulaire.

Les champs

Ajoutons maintenant les champs dont nous avons besoin pour la connexion : un champs de type *text* pour le login et un champs de type *password* pour le mot de passe, ainsi qu'un boutton de soumission.

```
<form method="post" action="login.php">


  <input type="text" name="login" required/>
  <input type="password" name="password" required/>


  <button type="submit">Connexion</button>


</form>
```

L'attribut `required` permet d'empêcher la soumission du formulaire si le champs n'est pas remplit. Chaque champs possède une attribut `name` pour l'identifier lors du traitement dans le scipt PHP.

Voilà notre formulaire est prêt !

Le script PHP

Créez maintenant un script `login.php` afin de traiter la soumission du formulaire.

Récupération des données

Afin de récupérer des données envoyées via une requête POST, il faut faire appel dans le code PHP à la variable globale `$_POST`. `$_POST` (comme `$_GET`) est un dictionnaire clé valeur (ou tableau associatif) qui contient les valeurs des données passées dans le formulaire, avec pour clé, l'attribut `name` du champs dans le formulaire.

Mais avant de récupérer les valeurs, il faut vérifier qu'elles sont bien présentes grâce à la fonction `isset` :

```
if(!(isset($_POST["login"]) && isset($_POST["password"]))){
  Header("Location: index.html");
}
```

Si ce n'est pas le cas, on utilise `Header` pour rediriger vers le formulaire. On peut ensuite récupérer les valeurs des champs à partir de leurs noms :

```
$login = $_POST["login"];
$password = $POST["password"];
```

Après avoir récupéré ces données, on va pouvoir les valider, mais pour cela, il faut se connecter à une base de donnée !

Connexion à une base de donnée

Commencez par créer une table et des données sur votre base de donnée en exécutant le script suivant sur datagrip :

```
CREATE TABLE utilisateur (
  login VARCHAR(255),
  password VARCHAR(255)
);
INSERT INTO utilisateur(login,password) VALUES ("JohnTest","Test@2020");
```

Pour se connecter à une base de donnée MySQL avec PHP, il faut utiliser PDO :

```
$hostname = "localhost";
$base = "nom de la base de donnée";
$loginBD = "root";
$passBD = "mot de passe de la base de donnée";

$bdd = new PDO ( "mysql:server=$hostname; dbname=$base", "$loginBD", "$passBD", array
(PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8'));
```

PDO (PHP Database Object) est orienté objet, c'est pour cela qu'on l'instancie avec le mot clé `new` et qu'on va devoir utiliser l'opérateur `->` dessus.

Pour éviter les injections SQL, il faut utiliser une requête préparée. Cela se fait avec la fonction `prepare` en utilisant des étiquettes dans la requête SQL :

```
$requete = $bdd->prepare ( "SELECT * FROM utilisateur WHERE login=:login AND password=:password" );
```

On peut ensuite lier les paramètres avec la méthode `bindParams` :

```
$requete->bindParam ( ":login", $login );
$requete->bindParam ( ":password", $password );
```

Et enfin, on exécute la requête :

```
$requete->execute ();
```

On peut ensuite récupérer la première ligne du résultat de la requête sous la forme d'un dictionnaire clé valeur (ou tableau associatif) avec la méthode `fetch()` :

```
$resultat = $resultatRequete->fetch();
```

Si la requête ne contient plus aucune ligne, on obtiens `false`. On peut donc vérifier que notre utilisateur est valide en faisant :

```
if($resultat){
  echo "Vous êtes connecté";
}
else {
  echo "Mauvais identifiants";
}
```

Voilà votre page de connexion est fonctionnelle, félifications

Revision #1
Created 21 November 2024 14:19:27 by Nicolas
Updated 21 November 2024 14:19:36 by Nicolas