IMT Nord Europe
École Mines-Télécom
IMT-Université de Lille

Luc Fabresse
luc.fabresse@imt-nord-europe.fr
CERI Systèmes Numériques

PHP

– Forms - GET/POST - Cookies/Session - SGBD –

16 mars 2023

Conserver ou transmettre des données entre deux requêtes HTTP

Le protocole HTTP est sans état c'est-à-dire que deux requêtes HTTP sont indépendantes et sans contexte. Lorsqu'un serveur Web reçoit une requête, il est impossible de savoir si l'utilisateur a déjà effectué une requête (par exemple s'il est déjà authentifié, ...), ... Il faut donc utiliser des techniques au dessus de HTTP afin de :

— Conserver l'identité de l'utilisateur connecté
— Déterminer l'identifiant de la page courante
— Connaître les valeurs des formulaires
— ...

Dans la suite, nous allons expérimenter quatre méthodes différentes. Chacune présente des avantages et inconvénients, il faut faut donc choisir la méthode adéquate à chaque problème. Committez tous les exercices suivants dans votre dépôt git dans un dossier nommé `TP3`.

1 Les formulaires en GET

1. Créer un fichier `login.php` avec le contenu suivant :

```
<form id="login_form" action="connected.php" method="GET">
  <table>
    <tr>
      <th>Login :</th>
      <td><input type="text" name="login"></td>
    </tr>
    <tr>
      <th>Mot de passe :</th>
      <td><input type="password" name="password"></td>
    </tr>
    <tr>
      <th></th>
      <td><input type="submit" value="Se connecter..." /></td>
    </tr>
  </table>
</form>
```

2. A travers un navigateur, essayez de vous connecter au site en remplissant ce formulaire.
3. Créer le fichier `connected.php` et afficher les valeurs des champs *login* et *password*
4. Ré-essayez de vous connecter au site, que remarquez-vous dans l'URL ?
5. Modifier le contenu du fichier `connected.php` ainsi :

```
<?php
  // on simule une base de données
  $users = array(
  // login => password
     'riri' => 'fifi',
     'yoda' => 'maitrejedi' );

  $login = "anonymous";
  $errorText = "";
  $successfullyLogged = false;

  if(isset($_GET['login']) && isset($_GET['password'])) {
```

```php
    $tryLogin=$_GET['login'];
    $tryPwd=$_GET['password'];

    // si login existe et password correspond
    if( array_key_exists($tryLogin,$users) && $users[$tryLogin]==$tryPwd ) {
       $successfullyLogged = true;
       $login = $tryLogin;
    } else
       $errorText = "Erreur de login/password";
  } else
    $errorText = "Merci d'utiliser le formulaire de login";


  if(!$successfullyLogged) {
      echo $errorText;
  } else {
      echo "<h1>Bienvenu ".$login."</h1>";
  }
?>
```

2 Les formulaires en POST

1. Le passage de valeurs en `GET` fait transiter les valeurs dans l'URL ce qui peut être problématique (taille des données, valeurs sensibles, ...). Modifier le code du formulaire ci-dessus pour utiliser le mode de passage `POST`
2. Ecrire le code PHP pour récupérer les valeurs entrées par l'utilisateur et les afficher (cf. `$_POST`)
3. La méthode `POST` est-elle plus "sécurisée" que `GET` ?

3 La gestion des *Cookies*

Les `cookies` sont des valeurs stockées dans le navigateur du client. Attention, les valeurs stockées dans les `cookies` ne doivent pas être sensibles du point de vue de la sécurité car elles sont accessibles et modifiables par l'utilisateur.

1. Intégrer le code suivant dans votre fichier `index.php` :

```php
<form id="style_form" action="index.php" method="GET">
  <select name="css">
    <option value="style1">style1</option>
    <option value="style2">style2</option>
  </select>
  <input type="submit" value="Appliquer" />
</form>
```

2. Intégrer du code PHP permettant de récupérer l'identifiant de style choisi par l'utilisateur et de le stocker dans un cookie en utilisant la fonction `setcookie`
3. Intégrer du code PHP qui permet :
   — de lire l'identifiant CSS dans les cookies (fixer un style par défaut si aucune valeur n'est présente dans les cookies)
   — d'inclure la bonne feuille de style dans la page HTML en fonction de cet identifiant (générer le code de la balise `link`)

Note : pourquoi l'union européenne et la RGPD a t-elle encadrée le stockage des cookies dans les navigateurs (i.e. un message demande à l'utilisateur d'accepter les cookies) ?

4 Le mécanisme de *Session*

Les variables de `Session` sont exactement comme des `Cookies` mais stockées côté serveur.

1. Lorsque l'utilisateur réussit à se connecter au site grâce au formulaire de login, ouvrir une session grâce à la fonction `session_start`
2. Enregistrer le login de l'utilisateur courant dans la session (`$_SESSION`)
3. Lorsqu'il est connecté, l'utilisateur doit pouvoir naviguer de page en page tout en restant connecté i.e. il faut afficher son login sur toutes les pages par exemple
4. Ajouter un lien pour se déconnecter. Il faut utiliser `session_unset` et `session_destroy`.

5 Sécurité et PHP

PHP permet d'exécuter du code sur un serveur et cela pose de nombreux problèmes de sécurité. Il est par exemple possible d'injecter du code PHP dans l'URL ou les formulaires ou même du code SQL.

Inscrivez-vous sur le site `www.root-me.org` et essayer de résoudre les premiers challenges Web Server :

`https://www.root-me.org/en/Challenges/Web-Server/`