Technologies Web

Farah Benamara Zitoune
Maître de conférences
IRIT-UPS
benamara@irit.fr

Plan du cours

- Cours 1 : Introduction HTML/CSS
- Cours 2 : Introduction programmation web + javascirpt
- Cours 3 : Introduction PHP
- Cours 4 : Php et Base de données
- Cours 5 : Php session+Cookies
- Si on a le temps : Introduction à Ajax et problèmes de sécurité

Transmettre des données entre des pages web

- Objectif : propager des variables entre les requêtes
  - authentification de visiteurs
  - garder des infos sur l'utilisateur tout au long de sa présence dans votre application
  - retenir/mémoriser des infos de formulaires d'une page à l'autre
  - informations de connexion
  - ...
- Les solutions
  - Les cookies
  - Les sessions

Les cookies

- Un cookie est un fichier stocké sur votre disque dur afin de permettre au serveur web de le reconnaître d'une page à l'autre

- Utilisé par de nombreux sites commerciaux pour conserver les préférences (ou profils) des utilisateurs exprimées (le plus souvent) dans des formulaires

- Les cookies ne peuvent pas collecter des infos sur le système utilisateur

Fonctionnement des cookies

- Ce sont des lignes particulières de l'entête HTTP qui permettent de véhiculer des informations entre client et serveur
  - Fonction setcookie qui doit être appelée avant toute balise <HTML> ou <HEAD>
  - Syntaxe:

Setcookie(string nom, string valeur, int exprire, string domain, string secure)

Les cookies

Setcookie(string nom, string valeur, int exprire, string domain, string secure)

- Tous les arguments sont optionnels sauf le nom.
- Les informations sur le domaine permettent de restreindre l'accès au cookie.
- Les informations sur l'expiration, permettent de restreindre la durée de vie du cookie, il est donc possible de déclarer un cookie qui ne restera sur l'ordinateur du visiteur qu'une certaine durée.

- Le paramètre de sécurité permet de restreindre l'envoi du cookie, ce qui permet de ne faire envoyer le cookie qu'aux serveurs sécurisés (HTTPS). Si le paramètre est désactivé, le cookie sera envoyé dans tous les cas.

Les cookies

Quelques exemple avec setcookie

```
<?php
 setcookie(“TestCookie”, “valeur de test”);
 setcookie(‘testCookie”, $value,  time()+3600); //expire dans une heure
 setcookie(‘testCookie”, $value, time()+3600, “/~ben/” , ”.ups-tlse.fr’’, 1)
 /*le cookie est valable pour toutes les pages dont le chemin commence par /~ben/ sur toutes les machines serveur dont le nom se finit par:  .ups-tlse.fr */
?>
```

Lecture du cookie

```
<?php   echo $_COOKIE[“TestCookie”]; ?>
```

Suppression du cookie

```
<?php   setcookie(“TestCookie”); /*on laisse que le nom du cookie*/ ?>
```

Les cookies :

Quelques exemple avec setcookie

```
<?
//affecte la valeur du cookie à $n
if (isset($_COOKIE["nb1"]))
{
setcookie("nb1",$_COOKIE["nb1"]+1);

}
else
{
setcookie("nb1",1);
}
echo 'Vous avez effectué'. $_COOKIE["nb1"].' visite(s)';

?>
```

Les cookies : un exemple complet

```
<!DOCTYPE html>
<?php
$cookie_name = "user";
$cookie_value = "John Doe";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/");
// 86400 = 1 day
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
      echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
      echo "Cookie '" . $cookie_name . "' is set!<br>";
      echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

<p><strong>Note:</strong> You might have to relaod the page to see
the value of the cookie.</p>

</body>
</html>
```

Les cookies

- Intérêts :
  - on ne transmet les variables que lorsqu'elles changent. Les cookies sont donc accessibles qu'au chargement de la prochaine page ou au rechargement de la page courante.
  - on ne fait pas de ré-écriture des formulaires et des liens

- Limitations :
  - 4Ko par cookie
  - 300 cookies par navigateur et 20 cookies par serveur (pour un même client)
  - Certains navigateurs ne supportent pas les cookies
  - Les utilisateurs peuvent les refuser

Les sessions

- Affectation d'un ID unique
  - pour chaque visiteur non connu (sans ID)
  - génération aléatoire de l'identifiant
- Les variables de session sont stockées sur le serveur dans le fichier: session.save_path (configuration php)
- A l'aide de ID session, php retrouve la valeur des variables session dans ce fichier

Fonctions liées aux sessions

- **session_start():** démarrer une session, de préférence en début de fichier
- **$_SESSION:** tableau global contenant toutes les variables session sourante
- **session_id():** revoie l'identifiant de session utilisé
- **session_destroy():** détruit une session en cours. Mais ne détruit pas les varaibles de session courante.
  - $_SESSION=array(); pour détruire le tableau des variables session

Pour une initiation complète, voir le lien :

www.phpdebutant.org/article69.php

Les sessions un exemple complet

```
<html>
 <head>
  <title>Connexion au site</title>
 </head>
 <body>
  <form method="post" action="verifLogin.php">
   <table border="0" width="400" align="center">
    <tr>
     <td width="200"><b>Vôtre login</b></td>
     <td width="200">
      <input type="text" name="login">
     </td>
    </tr>
    <tr>
     <td width="200"><b>Vôtre mot de passe<b></td>
     <td width="200">
      <input type="password" name="password">
     </td>
    </tr>
    <tr>
     <td colspan="2">
      <input type="submit" name="submit"
value="login">
     </td>
    </tr>
   </table>
  </form>
 </body>
</html>
```

```php
<?php
// On démarre la session
session_start();
$loginOK = false;  // cf Astuce

// On n'effectue les traitement qu'à la condition que
// les informations aient été effectivement postées
if ( isset($_POST) && (!empty($_POST['login'])) && (!empty($_POST['password'])) ) {
```

Récupération des variables du formulaires, $login=$_POST[login], …….

```php
  $sql = "SELECT pseudo, age, sexe, ville, mdp FROM user WHERE login = '".addslashes($login)."'";
  $req = mysql_query($sql) or die('Erreur SQL : <br />'.$sql);

  // On vérifie que l'utilisateur existe bien
  if (mysql_num_rows($req) > 0) {
     $data = mysql_fetch_assoc($req);

    // On vérifie que son mot de passe est correct
    if ($password == $data['mdp']) {
      $loginOK = true;
    }
  }
}

// Si le login a été validé on met les données en sessions
if ($loginOK) {
  $_SESSION['pseudo'] = $data['pseudo'];
  $_SESSION['age'] = $data['age'];
  $_SESSION['sexe'] = $data['sexe'];
  $_SESSION['ville'] = $data['ville'];
}
else {
  echo 'Une erreur est survenue, veuillez réessayer !';
}
?>
```

Les sessions un exemple complet

```php
<?php
// On appelle la session
session_start();

// On affiche une phrase résumant les infos sur l'utilisateur courant
echo 'Pseudo : ',$_SESSION['pseudo'],'<br />
     Age : ',$_SESSION['age'],'<br />
     Sexe : ',$_SESSION['sexe'],'<br />
     Ville : ',$_SESSION['ville'],'<br />';
?>
```

Les sessions un exemple complet

```php
<?php
// On appelle la session
session_start();

// On écrase le tableau de session
$_SESSION = array();

// On détruit la session
session_destroy();
?>
```

La redirection

- Pourquoi la redirection ?
  - changer les noms et adresses des pages sans perdre les liens existants
  - en cas d'erreur
  - en cas d'une authentification réussie

- Différentes méthodes
  - Au niveau du serveur: à l'aide de la fonction header
  - Au niveau du navigateur de l'internaute: méthode Meta Refreash

Redirection au niveau du serveur

- La méthode la plus propre

```
<?php
If ( $_SESSION[‘nom'] == " benamara")
{
  header("Location: https://"www.irit.fr");
  exit ;
}
?>
```

Redirection au niveau du navigateur

- A l'aide de la balise META

```
<html><head>
<meta http-equiv= "Refresh "  content= "0 ; URL=http://www.irit.fr " >
</head>
<body>
Cette page a été redirigée vers l'adresse: http://www.irit.fr
</body>
</html>
```

Nombre de secondes d'attente avant la redirection

o Certain navigateurs n'interprètent pas la balise META Refresh ➔ La redirection ne fonctionne pas toujours.