Les sessions php4
Beaucoup d'entre−vous se posent la question de savoir comment passer leur variables de pages en pages, ou encore comment conserver certaines informations pendant la durée d'une visite sur leur site. Les sessions devraient répondre à leurs attentes.

1 . Quelques petites choses à savoir sur les sessions

Une session est en fait un fichier conservé sur le serveur et accessible à vos scripts en fonction d'un identifiant généré à la création. Chaque fois qu'un de vos visiteurs génère une session, un identifiant lui est attribué. Tout ce qui est dans cette session est accessible de partout à vos scripts. On comprend dès lors très vite tout l'intérêt de la chose. En effet, si chaque session est propre à un visiteur, on peut personnaliser nos scripts en fonction du visiteur, ou encore alléger nos requêtes : plutôt que d'aller chercher un pseudonyme dans la base de donnée sur chaque page, vous le rapatriez à l'arrivée du visiteur sur le site, vous le stockez en session et c'est de là que vous y accéderez par la suite.

La session est finalement l'équivalent d'un cookie en plus sécurisé. En effet, étant stocké sur le serveur et non chez le client, elle est plus difficile d'accès aux éventuels pirates. Mais un risque demeure, prenez donc l'habitude de ne pas y stocker des informations trop sensibles. N'oubliez pas également que la session, à la différence du cookie, n'est valable qu'un temps limité (aux alentours des 30 minutes, mais cela dépend de la configuration de votre serveur), et est automatiquement détruite à la fermeture du navigateur du visiteur.

Vous pouvez enregistrer en session tout type de variable : du simple numérique au tableau en passant par la chaîne de caractères.

Il existe plusieurs fonctions liées aux sessions, mais deux seulement sont essentielles à leur fonctionnement, ainsi qu'un tableau :

session_start();
Cette fonction sert à démarrer une session OU appeler la session existante. Elle doit donc être présente sur toutes les pages de votre site. A savoir que cette fonction ne tolère pas d'envoi au navigateur avant elle. Plus simplement il ne faut aucune sortie avant elle : pas de html, pas de echo. Prenez l'habitude de la placer au début du fichier, avant quoi que ce soit d'autre et tout se passera bien.

session_destroy();
Cette fonction détruit la session en cours. Mais elle ne détruit pas les variables de sessions associées à la session courante. Nous verrons plus loin comment détruire une session complètement et proprement.

$_SESSION
Il s'agit du tableau global contenant toutes les variables de sessions pour la session courante. Son utilisation est exactement identique aux tableaux habituels, à savoir :

Utilisation du tableau global $_SESSION

```
// Nous allons créer une variable de session appellée "nombre" et qui contient "1234" :
$_SESSION['nombre'] = 1234;

// Nous pouvons maintenant afficher la variable "nombre"
echo $_SESSION['nombre'];  // Ceci va afficher 1234
```

2 . Etude de cas concret

Nous allons maintenant étudier un cas concret à savoir une procédure de login. En effet c'est une cas répandu d'utilisation des sessions. Nous allons donc créer un formulaire qui permet au visiteur de se loguer, puis une page de vérification qui va consulter la base de données et vérifier que les informations de connexion sont correctes. Et enfin mettre les données utiles sur l'utilisateur en session afin de pouvoir les réutiliser sur d'autres pages du site.

Le code HTML de la page de connexion

```
<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>
```

Le rendu

Votre login

Votre mot de passe

Rien à dire ici, si vous ne comprenez pas cette étape, il va falloir commencer par apprendre le HTML (quoique là le code en question n'est pas fantastique, mais c'est pour l'exemple :p). Nous allons maintenant passer à la récupération des données et au test de celles−ci.

Code de la page verifLogin.php

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

```php
// 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'])) ) {

  extract($_POST);  // je vous renvoie à la doc de cette fonction

  // On va chercher le mot de passe afférent à ce login
  $sql = "SELECT pseudo, age, sexe, ville, mdp FROM user WHERE login = '".addslashes($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 !';
}
?>
```

Il n'y a pas grand chose à expliquer ici. Si vous ne comprenez pas quelque chose, c'est qu'il vous faut relire les tutos précédents. Donc si l'utilisateur est logué, nous avons ses informations à disposition partout sur le site.

Astuce : la petite astuce que j'ai utilisée est en fait l'inverse du principe d'optimisme. Ce dernier veut qu'on part d'un constat qui dit que tout va bien et en cas de problème on gère l'erreur. Ici c'est le contraire. On part de l'idée que le login ne fonctionne pas en initialisant une variable à **false**. Et c'est seulement lorsque le login est confirmé qu'on modifie cette variable à **true**. Il suffit ensuite de tester cette variable pour loguer l'utilisateur ou pas. Cela nous économise des **else** inutiles à chaque étape de test.
Note : Remarquez que je n'ai effectué que les tests minimums sur le login avant de lancer la requête. Il conviendrait d'effectuer plus de tests afin d'éviter d'éventuels tentatives d'intrusions dans votre application. Mais ce n'est pas le sujet étudié ici. De même que pour la base de données, je vous épargne la création de la table, il s'agit d'une bête table à 5 colonnes...

Imaginons maintenant que nous voulions afficher ces informations plus tard, sur une autre page. Sans les sessions nous aurions dû faire une requête qui retournerait chercher le tout, tandis que là, il nous suffit de faire ça :

Code de la page affichInfo.php

```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 />';
?>
```

Notez que le **session_start()** est toujours présent. En effet il faut préciser au script que nous allons utiliser une session. C'est l'interpréteur qui va déterminer s'il faut en démarrer une ou sélectionner celle existante.
Bien entendu ce script n'a qu'une utilté réduite, à vous de voir quelles pourraient être les multiples façons de profiter des sessions.

3 . Déconnexion

Il s'agit là d'un point important. En effet une session reste ouverte tant que le visiteur ne ferme pas son navigateur. Vous devez lui offrir la possibilité de se déloguer d'une autre manière. Il vous suffit en fait de faire un lien appelé par exemple "déconnexion" qui pointe sur un script contenant ceci :

Code de la page logout.php

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

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

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

Voilà vous connaissez l'essentiel sur les sessions. Faites−en bon usage...