

Programmation Web

17 octobre 2017

Les sessions PHP (1/3)

- Elles permettent de stocker des variables de sessions qui seront connues dans toutes les pages qui lancent le système des sessions.
- On peut créer autant de variables de sessions qu'on veut sous la forme de clé/valeur dans le tableau \$_SESSION

```
<?php
//Lancement du système de sessions
session_name('pxxxxxx');
session_start();
//création et affectation dans une variable de
    session toto
$_SESSION['toto'] = 1;
?>
```

- \$_SESSION est un tableau de variables superglobales qui persiste sur le serveur tant que la session n'est pas terminée. Pour pouvoir y accéder, il faut démarrer le système de session dans la page.
- session_start doit forcément être appelé en tout début de page.

- `session_start()` démarre le système de session et si le visiteur vient d'arriver un identifiant de session (PHPSESSID) est généré. Il sera transmis de page en page grâce à un cookie ou à défaut peut être transmis via l'URL.

Les sessions PHP (3/3)

```
<?php
    session_name('pxxxxxx');
    session_start();
    // Delete the session cookie
    if (ini_get("session.use_cookies")) {
        $params = session_get_cookie_params();
        setcookie(session_name(), '', -1,
            $params["path"], $params["domain"],
            $params["secure"], $params["httponly"]
        );
    }
    // Détruit toutes les variables de session
    $_SESSION = array();
    // Détruit la session.
    session_destroy();
?>
```

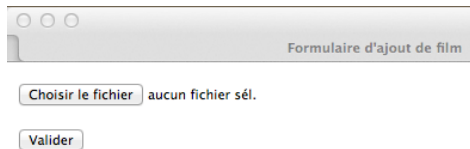
- En pratique, les sessions sont automatiquement détruites au bout d'un certain temps d'inactivité ou si vous quittez votre navigateur.

Les cookies

- Ce sont des informations qu'on peut stockées temporairement dans un fichier sur le poste du client
- Création d'un cookie : `setcookie` (name, value, expire, path, domain, secure, `httpOnly`)
 - `expire` est une date d'expiration exprimée en nombre de secondes depuis le 1er janvier 1970. C'est l'unité utilisée par la fonction `time` d'UNIX
 - `secure` rend le cookie inaccessible si la connexion n'est pas sécurisée (HTTPS)
 - `httpOnly` rend le cookie inaccessible à d'autres protocoles que le HTTP
- Le cookie n'est accessible qu'au prochain chargement d'une page.

```
<?php
$value = 'toto';
    // Création du cookie qui expirera dans une heure
    // accessible en HTTP seulement
    setcookie("TestCookie", $value, time()+3600, null,
        null, false, true);
?>
```

Formulaires et upload de fichiers



The screenshot shows a web browser window with a title bar containing three window control buttons. The page title is 'Formulaire d'ajout de film'. Below the title bar, there is a button labeled 'Choisir le fichier' followed by the text 'aucun fichier sél.'. Below this, there is a button labeled 'Valider'.

```
<FORM action="traitementFormFilm.php" method="POST"
      enctype="multipart/form-data">
  <input type="file" name="mon_fichier" />
  <input type="submit" value="Valider">
</FORM>
```

Le fichier est dans un dossier temporaire sur le serveur. Si on souhaite le garder, il faudra le placer ailleurs (dans un dossier du site par exemple). Si au contraire on décide de ne pas garder le fichier, il suffit de ne pas le déplacer ; il sera effacé automatiquement lorsque le script de traitement aura fini d'être exécuté.

Le tableau de superglobales \$_FILES

- `$_FILES['mon_fichier']['name']` contient le nom du fichier
- `$_FILES['mon_fichier']['type']` contient le type du fichier
- `$_FILES['mon_fichier']['size']` contient sa taille en octets
- `$_FILES['mon_fichier']['tmp_name']` contient l'emplacement temporaire du fichier
- `$_FILES['mon_fichier']['error']` contient le code d'erreur (0 si pas d'erreur ; 1 et 2 trop gros ; ...)
- `move_uploaded_file($_FILES['mon_fichier']['tmp_name'], $nouveau_nom)` permet de déplacer un fichier téléchargé par PHP. Le dossier de destination doit avoir les droits 777.

Les expressions régulières

- POSIX ou PCRE (+ rapide et performantes) pour PHP

- métacaractères :

^ \$ () [] { } ? + * . | \

- délimiteurs

/ % +

- l'option i après le délimiteur de fin de regex pour ignorer la casse
- les fonctions PCRE preg_match, preg_replace, ...

```
if(preg_match('#^[a-zA-Z]+.txt$#', $_FILES['  
    nom_fichier']['name']))
```


- Recommandées à partir de PHP 5.5
 - string **password_hash(string \$password)** Crée une clé de hachage pour un mot de passe stocké dans une chaîne de 255 caractères.
 - boolean **password_verify(string \$password, string \$hash)** vérifie qu'un mot de passe correspond à la version hachée.
- Pour le TP (PHP 5.3.3)
 - **sha1** (string) crée un mot de passe pouvant être stocké dans une chaîne de 40 caractères.

Le contrôle d'accès avec Apache (1/2)

- Contrôle de l'accès à un dossier avec login/mot de passe
- Rien à voir ici avec PHP
- Fait intervenir deux fichiers : .htaccess qui doit être dans le répertoire dont l'accès est contrôlé et .htpasswd
- le htaccess peut également être utilisé pour limiter ou interdire l'accès à certains numéros IP, certains utilisateurs ou certains groupes.

Exemple pour le fichier .htaccess :

```
AuthName "Connexion administrateur requise"  
AuthType Basic  
AuthUserFile "/home/UNIV-LYON1/isabelle.goncalves/  
    public_html/TP3/Admin/.htpasswd"  
Require valid-user
```

- Pour trouver le chemin absolu pour un fichier sur un serveur

```
<?php echo realpath("index.php"); ?>
```

Le contrôle d'accès avec Apache (1/2)

Exemple de fichier .htpasswd

```
fred:$1$nRSP5U.A$e8FqI6QTq/Bp6lNMjBUM01  
jul:$1$riMIdCaV$6G024RT5v4iwrSzChZq720  
monuser:$apr1$MWZtd0xs$mRBeIn.alFLzJZe4.r07U1
```

OU

```
fred:kangourou  
jul:totolitoto  
monuser:monpass
```

- Pour crypter un mot de passe : l'utilitaire htpasswd avec l'option -n Username
- [http ://shop.alterlinks.com/htpasswd/passwd.php](http://shop.alterlinks.com/htpasswd/passwd.php)