Programmation Orienté Objet

PHP – Gestion des fichiers

I. Ouverture d'un fichier

- Ouverture en lecture seulement depuis le début du fichier ("r") : `$fp = fopen(string fichier, "r");`
- Ouverture en écriture seulement depuis le début du fichier ("w") : `$fp = fopen(string fichier, "w");`
- Ouverture en écriture seulement depuis la fin du fichier ("a") : `$fp = fopen(string fichier, "a");`

```php
$fp = fopen("/home/rasmus/file.txt", "r");
$fp = fopen("../exemple.txt","a");
$fp = fopen("http://www.php.net/", "r");
$fp = fopen("ftp://user:password@example.com/", "w");
```

Il faut garder à l'esprit que l'on définit ici les fichiers au sens général du terme, vous pouvez donc grâce à ces instructions manipuler des fichiers *html*, *txt*, *php* etc.

`$fp` étant une variable, son nommage est bien entendu libre; il s'agit ici de la convention adoptée dans la documentation officielle PHP.

II. Les instructions principales

II.1. Ecriture dans un fichier

La fonction `fputs()` permet de lire un fichier; elle reçoit 2 arguments obligatoires et troisième faculatif `fputs($fp, $str, length);` :

- `$fp` : pointe sur le numéro de fichier ouvert par `fopen()`
- `$str` : représente la variable à enregistrer
- `length` : 3ème argument, faculatif, qui représente la longueur de la variable

Exemple

```php
// On déclare une variable dont la valeur (contenu) sera écrite dans le fichier
$myVar = "Bonjour le monde";

// Ouverture en écriture seule
$fp = fopen("essai.txt", "a");

// Ecriture du contenu ($myVar)
fputs($fp, $myVar);

// Fermeture du fichier
fclose($fp);
```

II.2. Lecture dans un fichier

La fonction `fgets()` permet de lire un fichier; elle reçoit 2 arguments : `fgets($fp, length);` :

- `$fp` : pointe sur la ressource de fichier ouvert avec `fopen()`
- `length` : représente la longueur d'enregistrement à lire (en octets)

Exemple :

```php
// Ouverture en lecture seule
$fp = fopen("essai.txt", "r");

// Boucle jusqu'à la fin du fichier
while (!feof($fp))
{
    // Lecture d'une ligne, le contenu de la ligne est affecté à la variable $ligne
    $ligne = fgets($fp, 4096);
    echo $ligne."<br>";
}
```

En fait l'instruction `fgets()` lit la ligne jusqu'à ce qu'elle rencontre un caractère de retour à la ligne `\n`. Par sécurité il est préférable de lui indiquer de lire 4096 caractères pour qu'elle puisse lire une ligne entière. Pour lire l'ensemble d'un fichier, vous pouvez aussi utilisez la fonction `file()`.

III. Les autres fonctions de gestion de fichiers

III.1.a. La fonction **basename()**

Cette fonction retourne le nom d'un fichier dans un URL ou un chemin d'accès à un fichier

Exemple :

```php
$path = "/home/httpd/html/index.php";
$file = basename($path); // $file retourne "index.php"
```

III.1.b. La fonction **copy()**

Copie un fichier vers un autre :

```php
$fichier ="toto.txt";
/* création d'un fichier toto.txt.bak */
copy($fichier, $fichier.'.bak');
```

III.1.c. La fonction **dirname()**

Retourne le répertoire (*directory*) dans lequel se trouve le fichier :

```php
$path = "/etc/passwd";
$file = dirname($path); /* $file retourne "/etc" */
```

IV. Exercice : un compteur texte en PHP

Les compteurs offerts par votre provider ne vous plaisent pas ?

Vous voulez avoir un compteur différent pour toutes les pages de votre site ?

Voici un petit script en PHP qui fera l'affaire :

```php
<?php
// On ouvre le fichier moncompteur.txt
$fichier = fopen("moncompteur.txt","r+");

// on lit le nombre indiqué dans ce fichier dans la variable
$visiteurs = fgets($fichier,255);

// on ajoute 1 au nombre de visiteurs
$visiteurs++;

// on se positionne au début du fichier
fseek($fichier,0);

// on écrit le nouveau nombre dans le fichier
fputs($fichier,$visiteurs);

// on referme le fichier moncompteur.txt
fclose($fichier);

// on indique sur la page le nombre de visiteurs
print("$visiteurs personnes sont passées par ici");

```

Quelques précisions et explications :

- Vous devez placer sur votre site un fichier *moncompteur.txt* avec juste le chiffre 0 dedans. Bien entendu si vous désirez que votre compteur démarre à 1254, vous pouvez le faire.
- La fonction `fopen()` permet d'ouvrir un fichier présent sur votre site. L'attribut *r+* permet de l'ouvrir en lecture et écriture.
- La fonction `fgets()` permet de lire le texte écrit dans le fichier. Le nombre 255 permet de ne lire que la première ligne, mais on pourrait également mettre 4 si l'on sait que le compteur ne dépassera pas 9999.
- `fseek()` permet de se repositionner au début du fichier. Ainsi, lorsque l'on réécrit le nouveau nombre de visiteurs, on est sûr d'effacer l'ancien.
- `fputs()` permet de réécrire la variable incrémentée dans le fichier.
- `fclose()` permet de refermer le fichier *moncompteur.txt* que l'on a ouvert au début du script.

Si vous désirez ajouter un compteur différent par page, vous n'avez qu'à recopier ce script sur vos différentes pages, en changeant juste *moncompteur.txt* en *moncompteur2.txt*, *moncompteur3.txt* etc. et en créant autant de fichiers *.txt* associés.

V. Téléchargement de fichier en PHP

Les formulaires HTML offrent la possibilité de télécharger un fichier : par exemple une photo, la notice d'un produit, un C.V. sur un site d'emploi (donc au format Word ou PDF).

Tout d'abord, un point sur le mot *télécharger* qui peut désigner aussi bien les opérations suivantes :

- enregistrement sur un PC d'un fichier présent sur un serveur distant (site web); il s'agit de **download**.
- envoi vers un serveur distant d'un fichier qui se trouve sur un PC : il s'agit de **l'upload**.

C'est ce second cas qui nous intéresse ici.

V.1. Formulaire HTML

Pour que le téléchargement soit possible, il faut ajouter l'attribut `enctype` à la balise `<form>`. La valeur doit être `multipart/form-data` :

```
<form action="post.php" method="post" enctype="multipart/form-data">
```

Ensuite, on a besoin d'un champ de type `file`, qui fera apparaître un bouton contenant le texte *Parcourir* avec lequel on pourra choisir un fichier présent sur le PC :

```
<input type="file" name="fichier">
```

V.2. Traitement en PHP

Dans le fichier de traitement PHP (celui assigné comme valeur à l'attribut `action`, lorsque le formulaire est soumis, on récupère les informations sur le fichier via la variable superglobale `$_FILES`. Comme les 7 autres superglobales, cette variable se comporte comme un tableau PHP.

Exercice : créer un formulaire d'upload et le fichier PHP de traitement correspondant, dans le fichier PHP écrivez juste `var_dump($_FILES);` et observez le résultat.

Vous devriez avoir quelque chose du genre :

- 'name' => string 'monfichier.jpg' (length=10) **= nom du fichier d'origine**
- 'type' => string 'image/jpeg' (length=10) **= type MIME du fichier**
- 'tmp_name' => string 'C:\wamp\tmp\phpC1CD.tmp' (length=23) **= nom et chemin du fichier temporaire**
- 'error' => int 0 **= erreurs (s'il y en a, elles sont retournées via un tableau PHP)**
- 'size' => int 100813 **= taille du fichier, en octets**

V.3. Gestion des erreurs

Si le téléchargement échoue, les erreurs sont retournées dans $_FILES["fichier"]["error"], les cas d'erreur sont prédéfinis dans un tableau : voir cette ressource.

S'il n'y a pas d'erreur, $_FILES["fichier"]["error"] vaut 0,

S'il y en a, la taille du tableau est supérieure à 0

Pour savoir si le tableau contient des erreurs, il faut donc faire :

```
if (sizeof($_FILES["fichier"]["error"]) > 0) { ... }
```

V.4. Sécurité

Le problème principal de l'upload d'un fichier est la sécurité : c'est l'utilisateur qui envoie un fichier présent sur son PC, et comme il ne faut jamais faire confiance aux actions de l'utilisateur, il faut vérifier que le fichier reçu est bien du type attendu et ne comporte pas de code malicieux (tentative d'attaque via un fichier exécutable, virus, logiciel espion etc.).

Il faut ensuite s'assurer des droits sur ce fichier (écriture, lecture, exécution) et le stocker correctement sur le serveur (s'agit-il d'un fichier accessible publiquement à tous les internautes ou d'un contenu confidentiel ?).

V.4.a. Vérifier le type

On doit tout d'abord s'assurer de points basiques :

- un fichier a-t-il bien été téléchargé ?
- le type du fichier envoyé par l'utilisateur est-il celui attendu (image, document Word, PDF...) ?

Les fausses bonnes idées, car les informations retournées ne sont pas fiables :

- tester uniquement l'extension comme chaîne de caractère
- tester le type MIME retourné par le navigateur (celui dans `$_FILES["fichier"]["type"]`).

PHP fournit une extension nommée _FILE*INFO* qui fait référence en termes de sécurité. Voici comment l'utiliser, pour un type :

```php
// On met les types autorisés dans un tableau (ici pour une image)
$aMimeTypes = array("image/gif", "image/jpeg", "image/pjpeg", "image/png", "image/x-png", "image/tiff");

// On extrait le type du fichier via l'extension FILE_INFO
$finfo = finfo_open(FILEINFO_MIME_TYPE);
$mimetype = finfo_file($finfo, $_FILES["fichier"]["tmp_name"]);
finfo_close($finfo);


if (in_array($mimetype, $aMimeTypes))
{
    /* Le type est parmi ceux autorisés, donc OK, on va pouvoir
       déplacer et renommer le fichier */

}
else
{
   // Le type n'est pas autorisé, donc ERREUR

   echo "Type de fichier non autorisé";
   exit;
}
```

V.4.b. Déplacer et renommer le fichier

Par défaut, le fichier téléchargé est stocké dans le répertoire `tmp` de votre serveur (sous Wamp dans `C:/wamp/tmp`). Mais ce fichier devra sans doute se trouver dans un répertoire de votre projet, il faut donc le déplacer.

Il est nécessaire aussi de renommer le fichier, pour répondre à une éventuelle charte de nommage et surtout pour que l'utilisateur ne puisse tenter d'exécuter le fichier via l'url (le nom sur le serveur sera différent de celui qu'il connaissait).

Pour cela, PHP propose une fonction "2 en 1" : `move_uploaded_file()`.

Exemple : déplacer et renommer un fichier (de type image) de `tmp` vers un répertoire nommé `images` d'un projet :

```php
move_uploaded_file($_FILES["fichier"]["tmp_name"], "images/photo.jpg");
```

La logique veut que les contrôles de sécurité ait été réalisés avant le déplacement.

V.5. La fonction glisser-déposer en HTML 5

HTML 5 propose une fonctionnalité de *glisser-déposer* (*Drag & Drop*) pour les fichiers. Il s'agit d'une API en Javascript. Exemple de mise en oeuvre.