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.

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">
```

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`, qui se comporte comme un tableau PHP.

On écrit `$_FILES["fichier"]`, `fichier` représente la valeur de l'attribut `name` du champ de type `file`.

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

Vous devriez obtenir quelque chose du genre :

Entrée	Description	Exemple de valeur
`$_FILES["fichier"]["name"]`	nom du fichier d'origine, sur votre PC	`string 'monfichier.jpg' (length=10)`
`$_FILES["fichier"]["type"]`	type MIME du fichier	`string 'image/jpeg' (length=10)`
`$_FILES["fichier"]["tmp_name"]`	nom et chemin du fichier temporaire	`string 'C:\wamp\tmp\phpC1CD.tmp' (length=23)`
`$_FILES["fichier"]["error"]`	int 0 = erreurs (s'il y en a, elles sont retournées via un tableau PHP)	`nom et chemin du fichier temporaire`
`$_FILES["fichier"]["size"]`	taille du fichier, en octets	int 100813

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, `$_FILES["fichier"]["error"]` retourne **un entier** valant 0,
- S'il y a des erreurs, `$_FILES["fichier"]["error"]` retourne **un tableau**. Pour afficher les erreurs, il faut donc la condition suivante pour lire les erreurs :

```
1  // Si c'est un tableau et que celui-ci n'est pas vide
2  if (is_array($_FILES["fichier"]["error"]) && !empty($_FILES["fichier"]["error"]) )
3  {
4      // Boucle pour afficher les messages d'erreurs
5  }
```

Les codes d'erreur correspondent aux éléments suivants.

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.

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 ?).

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"]`).

Exemple, le mauvais

- Copier l'un de vos fichiers CSS, renommez cette copie en changeant l'extension en 'jpg'.
- Chargez ce fichier dans votre formulaire
- Dans le fichier `post.php`, mettez le code suivant :

```
1  // Le code suivant récupère l'extension ('jpg')
2  $extension = pathinfo($_POST["fichier"]["tmp_name"], PATHINFO_EXTENSION);
```

> Regardez la documentation des fonctions `substr()` et `strrchr()` pour comprendre ce qu'elles font.

```
1  // Tableau des extensions autorisées par le site (ici des fichiers de type image)
2  $aAllowed = array("gif", "jpeg", "jpg", "png", "tiff");
3
4  // Teste que l'extension du fichier téléchargé est bien autorisée
5  if (in_array())
6  {
7     echo"Type de fichier autorisé.";
8  }
9  else
10 {
11    echo"Type de fichier non autorisé.";
12 }
```

Exemple, le bon

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

```
1  // On met les types autorisés dans un tableau (ici pour une image)
2  $mimeTypes = array("image/gif", "image/png", "image/pjpeg", "image/png", "image/x-png", "image/tiff");
3
4  // On ouvre l'extension FILE_INFO
5  $finfo = finfo_open(FILEINFO_MIME_TYPE);
6
7  // On extrait le type MIME du fichier via l'extension FILE_INFO
8  $mimetype = finfo_file($finfo, $_FILES["fichier"]["tmp_name"]);
9
10 // On ferme l'utilisation de FILE_INFO
11 finfo_close($finfo);
12
13 if (in_array($mimetype, $aMimeTypes))
14 {
15    /* Le type est parmi ceux autorisés, donc OK, on va pouvoir
16       déplacer et renommer le fichier */
17 }
18 else
19 {
20    // Le type n'est pas autorisé, donc ERREUR
21
22    echo "Type de fichier non autorisé";
23    exit;
24 }
```

Déplacer et renommer le fichier

Par défaut, le fichier téléchargé est renommé avec un nom temporaire et stocké dans un répertoire nommé *tmp/* (pour temporaire) de votre serveur (*C:/laragon/tmp*, *C:/wamp/tmp*). Il faudra alors déplacer votre fichier de *tmp/* vers un répertoire final de votre choix (par exemple dans *images/*).

Il est nécessaire aussi de renommer votre fichier pour que l'utilisateur ne puisse tenter d'exécuter le fichier via l'url (ainsi 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 `tmp/` vers un répertoire nommé `images/` :

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

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

Dans votre projet, vous devez bien sûr remplacer *photo.jpg* par le nom de fichier souhaité, c'est-à-dire le pro_id et l'extension du fichier téléchargé. Le code suivant vous permettra d'obtenir l'extension :

```
$extension = pathinfo($_POST["fichier"]["tmp_name"], PATHINFO_EXTENSION);
```

ou, alternative :

```
$extension = substr(strrchr($_FILES["fichier"]["name"], "."), 1);
```

Pour aller plus loin

Limite de poids des fichiers

Le chargement des fichiers sur un serveur, et en particulier la taille des fichiers uploadés, est soumis à plusieurs limitations au niveau de la configuration dudit serveur.

Ces paramètres limitatifs sont modifiables dans le fichier de configuration de PHP, *php.ini* (situé dans *C:/laragon/bin/php/php-X.X.X-Win32-VC15-x64* ou *C:/wamp/bin/apache/apache2.X.XX.bin/*).

Les paramètres pris en compte pour maîtriser l'upload des fichiers sont les suivants :

- `upload_max_filesize` : fixe la taille maximale d'un fichier à charger, exprimée en octets.
- `post_max_size` : fixe le poids maximum des données envoyées par le formulaire (donc des fichiers chargés par formulaire sur le serveur). La valeur de `post_max_size` doit logiquement être supérieure à la valeur de `upload_max_filesize`.

Vérifiez aussi la valeur du paramètre `memory_limit` qui fixe la mémoire maximum qu'un script peut allouer à une requête. La valeur de `memory_limit` doit être supérieure ou égale à la valeur de `post_max_size`.

Pour connaître leur valeur, utilisez la fonction `ini_get()` :

```
echo ini_get("upload_max_filesize");
```

Pour la modifier, on peut utiliser la fonction `ini_set()`, à placer en haut de votre formulaire :

```
ini_set("upload_max_filesize", 104857600);
```

Limiter la taille du fichier à uploader au niveau du formulaire HTML

Il est possible de limiter, côté client, la taille des fichiers uploadés, **tout en gardant à l'esprit que la taille max fixée côté client ne peut pas être supérieure à celle fixée par le serveur**.

Pour fixer une taille limite dans le formulaire, il faut ajouter un champ caché **avant le champ de type `file`** :

```
<input type="hidden" name="MAX_FILE_SIZE" value="104857600">
```

- la valeur de l'attribut `name` doit être `MAX_FILE_SIZE`
- l'attribut `value` indique une valeur pour le poids maxi du fichier téléchargé. L'unité est l'octet. Ici, 104857600 octets correspond à 100 mégaoctets (il existe sur le net des convertisseurs).

> Comme tout code côté client, la valeur de `MAX_FILE_SIZE` peut être facilement modifiée dans la console navigateur.

Spécifier des droits sur le fichier

Sur les systèmes d'exploitation (Windows, Linux...), les fichiers possèdent des droits (ou permissions) de lecture, d'exécution et d'écriture accordés aux utilisateurs. Il s'agit d'un système un peu complexe mais qui participe grandement à la sécurité.

Lire ces ressources :

- Ressource 1
- Ressource 2

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.

Ressources

- php.net

Exercice

Dans le projet *Jarditou*, mettre en oeuvre le téléchargement de fichier dans le formulaire d'ajout pour pouvoir charger la photo d'un produit.

Consignes :

- Respectez la charte de nommage "pro_id.extension" (exemple : *1.jpg*).
- Ne pas mettre en oeuvre les notions de droits sur les fichiers.
- Ne pas mettre en oeuvre le Drag & Drop.