

Cours de PHP



Jean-Luc Collinet

Haute École Léonard de Vinci : Paul Lambin
Bloc 1 du Bac en Informatique

Au menu d'aujourd'hui...

- *Upload* de fichiers
- Gestion d'images
- Aspects de sécurité
- Développement versus Intégration

Upload de fichiers

- On va *uploader* un fichier depuis le poste client **vers le serveur Web**
- En situation réelle : localhost != serveur Web
- Le fichier *uploadé* est **à sauvegarder dans un répertoire de l'application Web**
- Référence à consulter
 - <http://www.php.net/features.file-upload>



Formulaire HTML gérant l'*upload*

<!-- Le type d'encodage des données enctype doit être spécifié -->

```
<form enctype="multipart/form-data" action="URL à préciser"  
method="post">
```

<!-- MAX_FILE_SIZE doit précéder le champ input de type file -->

```
<input type="hidden" name="MAX_FILE_SIZE"  
value="1000000" />
```

<!-- Le *name* est à bien définir, il détermine l'indice dans le tableau associatif \$_FILES -->

Envoyez ce fichier : <input type="file" name="userfile" />

```
<input type="submit" value="Envoyer le fichier" />
```

```
</form>
```

Du côté du serveur Web

- Le fichier *uploadé* est **copié automatiquement** dans un répertoire **temporaire**
- Le tableau associatif `$_FILES` est rempli :
 - `$_FILES['userfile']['tmp_name']`
 - Nom du fichier copié temporairement sur le serveur
 - `$_FILES['userfile']['name']`
 - Nom du fichier d'origine sur la machine client

Que faire du fichier *uploadé* ?

- Il faut copier ce fichier dans un des répertoires de votre site Web
- Fonction pour réaliser cette copie :
 - `move_uploaded_file($origine,$destination)`
- Dans le cadre du cours, gérons des fichiers images
 - À copier dans un répertoire images/
 - Chaque URL d'image est à stocker en base de données

Copier un fichier dans images/

```
$origine = $_FILES['userfile']['tmp_name'];
```

```
$destination = 'views/images/' .  
                $_FILES['userfile']['name'];
```

```
move_uploaded_file($origine,$destination);
```

En base de données

- Nous allons stocker
dans un champ de type texte
l'URL de l'image
qui est copiée dans notre architecture
- Exemples de chaînes stockées en table :
 - 'views/images/smile.jpg'
 - 'views/images/ouistiti.png'
 - 'views/images/1457691790_7832timestamp.gif'

Points à réfléchir

- Taille de l'image en bytes
 - `<input type="hidden" name="MAX_FILE_SIZE" value="1000000" />`
- Doublons possibles des noms de fichiers
 - Ajouter dans le nom d'un horodatage
- Vérifier le type du fichier ?!
 - `$_FILES['userfile']['type']`

Sécurité

- Attention aux failles de l'*upload*... !
 - Cf. document joint
 - Les Failles de l'Upload en PHP par Guillaume LAUMAILLET
- Vérifier l'extension du fichier
- Vérifier le mime-type avec `getimagesize()`

Authentification avancée

- Table d'utilisateurs en base de données
- Mot de passe utilisateur à hacher \equiv Hash grâce à des fonctions PHP
 - Md5 et Sha1 : déconseillées ! <http://md5decrypt.net/>
 - Hachage Blowfish avec Salt 😊
 - Fonctions PHP à utiliser :
 - password_hash (ou crypt)
 - password_verify
 - <http://php.net/manual/fr/faq.passwords.php>

Exemple de code...

- `$mdp = '@HanSolo2018/';`
- `$hash = password_hash($mdp, PASSWORD_BCRYPT);`
`$2y$10$ANpU4u5KF5wu1GQ8vchmAeVh31NmibZjgbJsVd38qsnbQHTODRr5a`
- `if (password_verify($mdp, $hash)) {`
 `$msg = 'Le mot de passe est valide.';`
 `}`

« Hash » selon Blowfish

- 60 caractères à stocker en DB

`$2y$10$6z7GKa9kpDN7KC3ICW1Hi.f`
`d0/to7Y/x36WUKNP0IndHdkdR9Ae3K`

Diagram illustrating the structure of a Blowfish hash string:

- Algorithm**: \$2y\$ (red line)
- Algorithm options (eg cost)**: 10\$ (blue line)
- Salt**: 6z7GKa9kpDN7KC3ICW1Hi.f (green line)
- Hashed password**: d0/to7Y/x36WUKNP0IndHdkdR9Ae3K (orange line)

Autre aspect de sécurité

- Sécurisation d'un répertoire
 - Au niveau du serveur Web Apache
 - Technique « .htaccess et .htpasswd »
- Une authentification éventuelle est ainsi gérée par le navigateur et non par le site Web
- Démonstration...

Pour du PHP professionnel...

- Développer ou Intégrer ?
- Développer à partir de rien devient rare
- Développer sur base d'un framework PHP généralement MVC OO 😊
- Intégrer c'est utiliser une solution (type CMS) déjà développée !
 - *Content Management System*
 - Système de Gestion de Contenu

Développer versus Intégrer

- Répondre aux besoins d'informatisation avec grande adéquation
 - Liberté totale
 - Possibilité aisée d'extensions
- Analyser les besoins
- Architecturer sur base d'un framework
- Coder 😊
- Documenter le code

Exemples de framework

- [http://fr.wikipedia.org/wiki/Liste de frameworks PHP](http://fr.wikipedia.org/wiki/Liste_de_frameworks_PHP)
- CakePHP
- CodeIgniter
- Laravel
- Zend Framework
- Symfony 4.2.3

Intégrer versus Développer

- Ne pas réinventer la roue
- Utiliser une solution toute faite
- Mise en œuvre **très rapide**
- Décoder une architecture pour l'adapter
 - Pas du tout évident !
- Idéal pour répondre à des **besoins génériques**
 - Une extension peut s'avérer difficile

Exemples de solutions libres

- Liste : www.opensourcecms.com/
- Joomla
 - <http://www.joomla.fr/>
- Drupal
- Moodle
 - <https://moodle.org/>
- osCommerce
 - <http://fr.wikipedia.org/wiki/OsCommerce>

Communauté Open-Source

- Le tout est « gratuit » !
- Monde de services plutôt que de produits
- Logiciels libres
 - http://fr.wikipedia.org/wiki/Logiciel_libre
- Licences
 - BSD, GPL, CeCILL, Creative Commons

Se procurer de l'aide



- Recherche d'une solution à un problème
 - Google It !
- Forum communautaire
 - <http://stackoverflow.com/>
- Collègues développeurs
- Tutoriels et exemples en ligne
- Livres en .pdf ou en version papier
 - <https://itbook.store/books/php>

Bonne évolution à tous



- Il n'y a pas de conclusion...
Vivre est un élan merveilleux,

Jean-Luc Collinet