Module 7
Gestion des fichiers

Contenu du module

- Fonctions PHP pour la manipulation des fichiers.
  - Lecture et écriture de fichiers.
  - Création et parcours de répertoires.
  - Les directives de configuration php.ini associées à la manipulation de fichiers.
- Téléchargement et envoi de fichiers.

Fonctions PHP pour la manipulation des fichiers

- PHP propose un grand nombre de fonctions permettant de manipuler les fichiers sur le serveur
  - Il s'agit par exemple de réaliser des opérations sur le système de fichiers
    - Copie, déplacement, suppression, création, …
  - Ou des opérations sur les fichiers et répertoires
    - Listing, lecture, écriture, …
- Une grande majorité de ces fonctions renvoient une ressource PHP
  - Elle doit être fermée après usage
- Considérations sur les plateformes
  - Sous Windows il faut échapper l'anti-slash (par un anti-slash = \\)
    - Ou bien utiliser une notation type "Unix", avec des slashs (/).
  - La constante prédéfinie DIRECTORY_SEPARATOR donne le caractère de séparation utilisé dans les noms de répertoire
  - La constante prédéfinie PHP_EOL donne la séquence de caractères utilisée par la plate-forme pour représenter une nouvelle ligne

Lecture et écriture de fichiers

- **`fopen`**
  - Ouvrir un fichier
    - `fopen(string $filename, string $mode, bool $use_include_path = false, ?resource $context = null): resource|false`
- **`fclose`**
  - Fermer un fichier
    - `fclose(resource $stream): bool`
- **`fread`**
  - Lire le contenu d'un fichier (dans une chaîne)
    - `fread(resource $stream, int $length): string|false`
- **`fwrite`**
  - Écrire dans un fichier
    - `fwrite(resource $stream, string $data, ?int $length = null): int|false`

- **`file`**
  - Lire le contenu d'un fichier (dans un tableau)
    - `file(string $filename, int $flags = 0, ?resource $context = null): array|false`

Lecture et écriture de fichiers – Fonctions avancées

- **`file_get_contents`**
  - Ouvrir, lire et fermer un fichier

```
file_get_contents(
    string $filename,
    bool $use_include_path = false,
    ?resource $context = null,
    int $offset = 0,
    ?int $length = null
): string|false
```

- **`file_put_contents`**
  - Ouvrir, écrire et fermer dans un fichier

```
file_put_contents(
    string $filename,
    mixed $data,
    int $flags = 0,
    ?resource $context = null
): int|false
```

Création et parcours de répertoires

- **copy**
  - Copier un fichier
    - `copy(string $from, string $to, ?resource $context = null): bool`
- **unlink**
  - Supprimer un fichier
    - `unlink(string $filename, ?resource $context = null): bool`
- **rename**
  - Renommer un fichier
    - `rename(string $from, string $to, ?resource $context = null): bool`
- **file_exists**
  - Tester l'existence d'un fichier ou d'un répertoire
    - `file_exists(string $filename): bool`
- **mkdir**
  - Créer un répertoire
    - `mkdir(string $directory, int $permissions = 0777, bool $recursive = false, ?resource $context = null): bool`
- **opendir**
  - Ouvrir un répertoire
    - `opendir(string $directory, ?resource $context = null): resource|false`
    - `Permet l'usage de readdir() et closedir() et rewinddir()`
- **closedir**
  - Fermer un répertoire
    - `closedir(?resource $dir_handle = null): void`
- **readdir**
  - `Lire le contenu d'un répertoire`
  - `readdir(?resource $dir_handle = null): string|false`

Les directives de configuration – `php.ini`

- **`open_basedir`** *`string`*
  - Limite les fichiers pouvant être accédés par PHP à une architecture de dossiers spécifique, incluant le fichier lui-même
  - Lorsqu'un script tente d'accéder à un fichier avec, par exemple, la fonction include ou la fonction fopen(), le chemin vers le fichier est analysé
    - Lorsque le fichier se trouve à l'extérieur de l'architecture de dossiers spécifié, PHP refusera d'y accéder
- **`include_path`** *`string`*
  - Spécifie une liste de répertoires où les fonctions require, include, fopen(), file(), readfile() et file_get_contents() chercheront les fichiers
    - Une liste de répertoires séparés par deux points (:) sous Unix ou par un point-virgule (;) sous Windows.
- **`file_uploads`** *`bool`*
  - Autorise ou non le chargement de fichiers par HTTP
    - Voir aussi les directives `upload_max_filesize`, `upload_tmp_dir` et `post_max_size`
- **`upload_tmp_dir`** *`string`*
  - Répertoire temporaire utilisé pour stocker les fichiers lors du chargement distant (upload)
    - L'utilisateur sous lequel fonctionne PHP doit avoir les droits en écriture sur ce répertoire
- **`upload_max_filesize`** *`int`*
  - La taille maximale en octets d'un fichier uploadé
- **`max_file_uploads`** *`int`*
  - Le nombre maximum de fichiers pouvant être envoyés simultanément

Téléchargement et envoi de fichiers - Télécharger

- Pour permettre de télécharger un fichier depuis une application Web, il suffit de faire un lien (balise HTML <a>) pointant sur ledit fichier
  - Mais c'est le navigateur qui décide de proposer à l'utilisateur un dialogue d'enregistrement ou d'afficher directement le document s'il sait comment faire
- Il est possible de forcer le téléchargement en utilisant la technique suivante :
  - Envoyer des entêtes spécifiques avec la fonction header()
    - `header("Content-Disposition: attachment; filename=...");`
      - `filename` est le nom de fichier qui sera proposé à l'enregistrement
    - `header("Content-Type: ...");`
      - Le type de contenu du fichier
  - Envoyer le contenu du fichier à télécharger
    - Avec la fonction `readfile()`
      - `readfile(string $filename, bool $use_include_path = false, ?resource $context = null): int|false`
  - Le lien de téléchargement devra pointer sur le script PHP contenant ces instructions

Téléchargement et envoi de fichiers - Envoi

- Plusieurs étapes pour mettre en place une fonctionnalité d'upload de fichiers sur un site ou une application PHP :
  - Le formulaire
    - Paramétrer le formulaire pour l'envoi de données binaires
    - Proposer une zone permettant à l'utilisateur de spécifier le fichier à transférer
  - Le script de traitement du formulaire
    - Récupérer le fichier envoyé par l'utilisateur
    - Appliquer les vérifications nécessaire
    - Manipuler le fichier
- Attention car ce type de fonctionnalité peut générer des failles de sécurité si elle n'est pas correctement mise en œuvre !

Formulaire d'envoi de fichier

- Paramétrage :
  - Pour provoquer le transfert du fichier, il faut ajouter l'attribut `enctype="multipart/form-data"` à la balise `<form>`
- Le formulaire peut contenir n'importe quel type de champ de formulaire HTML
  - Pour l'upload de fichier, le champ est de type `<input type="file" … />`
- Il est possible d'ajouter une zone cachée dans le formulaire afin de limiter la taille des fichiers qui peuvent être envoyés vers le serveur
  - Cette zone cachée, obligatoirement située avant la zone de type file doit s'appeler `MAX_FILE_SIZE` (attribut `name`) et préciser la taille maximum en octets dans l'attribut `value`
  - La valeur précisée dans cette zone ne peut pas être supérieure à la valeur de la directive de configuration `upload_max_filesize` (`php.ini`)
  - Si la zone cachée n'est pas présente, c'est la taille spécifiée dans la directive `upload_max_filesize` qui s'applique

Traitement de l'envoi de fichier

- Lorsqu'un fichier est envoyé avec un formulaire, des informations sur ce fichier sont accessibles dans le tableau $_FILES
  - La valeur saisie par l'utilisateur n'est plus disponible dans $_POST !
- $_FILES est un tableau associatif multidimensionnel
  - La première clé est égale au nom de la zone de type file du formulaire
  - La valeur est un tableau associatif contenant cinq clés :

Clé	Valeur
`name`	Nom du fichier (sans chemin d'accès)
`type`	Type MIME du fichier (fourni par le navigateur)
`size`	Taille du fichier en octets
`tmp_name`	Nom du fichier temporaire créé sur le serveur (chemin complet)
`error`	Code d'erreur. Une des constantes suivantes : UPLOAD_ERR_OK (0) : pas d'erreur UPLOAD_ERR_INI_SIZE (1) : taille du fichier supérieure à la taille définie par la directive de configuration `upload_max_filesize` UPLOAD_ERR_FORM_SIZE (2) : taille du fichier supérieure à la taille définie par l'option `MAX_FILE_SIZE` du formulaire UPLOAD_ERR_PARTIAL (3) : le fichier a été chargé partiellement UPLOAD_ERR_NO_FILE (4) : aucun fichier saisi UPLOAD_ERR_NO_TMP_DIR (6) : pas de répertoire temporaire UPLOAD_ERR_CANT_WRITE (7) : erreur lors de l'écriture du fichier sur disque UPLOAD_ERR_EXTENSION (8) : transfert stoppé par une extension

Manipulation du fichier transféré

- Il est nécessaire de commencer par vérifier la réussite du transfert
  - En regardant la clé `error` du tableau associé au champ du fichier
- Puis, il faut vérifier l'authenticité du fichier
  - Vérifier le type MIME
  - Vérifier l'extension
  - Mais cela n'est pas suffisant !
    - On peut compléter avec la fonction `is_uploaded_file()` permettant de vérifier que le fichier a bien été transféré en HTTP
    - Idéalement, il faut aussi ouvrir le fichier avec une bibliothèque dédiée au format et vérifier qu'il est exploitable
- Il est ensuite possible de manipuler le fichier sur le serveur
  - La fonction `move_uploaded_file()` est dédiée au déplacement du fichier uploadé du répertoire temporaire vers un répertoire du site ou de l'application Web

Travaux pratiques