OFPPT

M 107 : Développer des sites web dynamiques

1 ère année

Filière : Développement Digital (Tronc commun)

Partie 2: Programmer en PHP

Chapitre 2: Traiter les données en PHP

Manipulation de fichier : Ouvrir un fichier

La fonction **fopen()** en PHP est utilisée pour ouvrir un fichier en mode lecture, écriture ou d'autres modes d'accès. Voici une explication détaillée de sa syntaxe et des exemples d'utilisation.

Syntaxe

fopen(string $filename, string $mode, bool $use_include_path = false, resource $context = ?): resource

- **$filename :** Chemin du fichier à ouvrir.
- **$mode :** Mode d'ouverture (lecture, écriture, ajout etc.).
- **$use_include_path (optionnel) :** Si true, PHP cherchera le fichier dans les chemins définis par include_path définie dans le fichier php.ini.
- **$context (optionnel) :** Un contexte de flux pour configurer des options avancées (HTTP, SSL…)

Manipulation de fichier : Ouvrir un fichier

Exemple: `$idFile = fopen("test.txt", "w");`

Mode	Description
r	Ouvre le fichier en lecture seule. Le pointeur est placé au début du fichier. Si le fichier n'existe pas, l'ouverture échoue.
r+	Ouvre le fichier en lecture et écriture. Le pointeur est placé au début du fichier. Si le fichier n'existe pas, l'ouverture échoue. Permet de modifier le contenu existant.
w	Ouvre le fichier en écriture seule. Le pointeur est placé au début du fichier, et la taille du fichier est réduite à 0 (efface le contenu). Si le fichier n'existe pas, il est créé.
w+	Ouvre le fichier en lecture et écriture. Comme w, il écrase le fichier existant ou le crée s'il n'existe pas. Le pointeur est placé au début.
a	Ouvre le fichier en écriture seule. Le pointeur est placé à la fin du fichier (ajout). Si le fichier n'existe pas, il est créé. Idéal pour les logs.
a+	Ouvre le fichier en lecture et écriture. Le pointeur est placé à la fin pour l'écriture (ajout), mais au début pour la lecture. Si le fichier n'existe pas, il est créé.
x	Crée et ouvre le fichier en écriture seule. Le pointeur est placé au début. Si le fichier existe déjà, l'ouverture échoue
x+	Crée et ouvre le fichier en lecture et écriture. Comme x, mais permet aussi la lecture. Si le fichier existe déjà, l'ouverture échoue.
c	Ouvre le fichier en écriture seule. Le pointeur est placé au début. Si le fichier n'existe pas, il est créé. Contrairement à w, il ne tronque pas le fichier existant.
c+	Ouvre le fichier en lecture et écriture. Comme c, mais permet aussi la lecture. Le contenu existant n'est pas tronqué.

Manipulation de fichier : Fermer un fichier

fclose()

Ferme un fichier ouvert. **Toujours fermer un fichier après l'avoir utilisé !**

Syntaxe

fclose(resource $handle): bool

Exemple

```
$fichier = fopen("exemple.txt", "r");
fclose($fichier);
```

Explication

- Libère la mémoire occupée par le fichier.

Manipulation de fichier : Lire un fichier

fread() : Lire un fichier en mode binaire. La fonction **fread()** lit un certain **nombre d'octets** à partir d'un fichier ouvert.

Syntaxe

fread(resource $handle, int $length): string

- **$handle :** Le pointeur du fichier (obtenu via **fopen**()).
- **$length :** Nombre d'octets à lire.

Exemple : Lire tout un fichier

```
if (file_exists("test.txt")) {
    $idfile = fopen("test.txt", "r");
    if ($idfile) { // Vérifie si l'ouverture a réussi
        echo fread($idfile, filesize("test.txt"));
        fclose($idfile);
    } else {
        echo "Erreur : impossible d'ouvrir le fichier.";}}
```

Manipulation de fichier : Lire un fichier

file()

La fonction **file()** charge le fichier ligne par ligne dans un tableau.

Syntaxe

file(string $filename, int $flags = 0): array|false

Renvoie un **tableau** où chaque élément correspond à une ligne du fichier.

Exemple

```
$lignes = file("test.txt");
foreach ($lignes as $ligne) {
    echo $ligne . "<br>";
}
```

Manipulation de fichier : Lire un fichier

fstat()

Cette fonction retourne des **informations sur un fichier ouvert** (taille, permissions, type, etc.).

Syntaxe

fstat(resource $handle): array|false

Exemple

```
$handle = fopen("test.txt", "r");
$info = fstat($handle);
fclose($handle);
print_r($info);
```

Manipulation de fichier : Lire un fichier

fgets()
Lit **une seule ligne** à la fois.

Syntaxe
fgets(resource $handle, int $length = ?): string|false

Exemple

```
$handle = fopen("test.txt", "r");
while (($ligne = fgets($handle)) !== false) {
    echo $ligne . "<br>";
}
fclose($handle);
```

Explication
- **fgets()** lit une ligne à la fois.
- Boucle **while** pour lire jusqu'à la fin du fichier.

Manipulation de fichier : Lire un fichier

fgetc()

Lit **un seul caractère** à la fois.

Syntaxe

fgetc(resource $handle): string|false

Exemple

```
$handle = fopen("test.txt", "r");
while (($char = fgetc($handle)) !== false) {
   echo $char;
}
fclose($handle);
```

Explication

- fgetc() lit caractère par caractère.
- Peut être utilisé pour l'analyse de fichiers.

Manipulation de fichier : Lire un fichier

file_get_contents

lire **tout le contenu d'un fichier d'un coup, r**etourne une chaîne contenant tout le contenu du fichier.

Syntaxe

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

Exemple

```
$contenu = file_get_contents("test.txt");
echo $contenu;
```

Manipulation de fichier : Lire un fichier

readfile()

Cette fonction **lit** et **affiche** directement un fichier.

syntaxe

readfile(string $filename, bool $use_include_path = false, resource $context = ?): int|false

Exemple

```
readfile("test.txt");
```

- Affiche directement le contenu du fichier.
- Plus rapide que file_get_contents() si on veut juste afficher.

Manipulation de fichier : Ecrire dans un fichier

fwrite()

Écrit du texte dans un fichier.

Syntaxe

fwrite(resource $handle, string $data, int $length = ?): int|false

Exemple

```
$handle = fopen("test.txt", "w");
fwrite($handle, "Bonjour !");
fclose($handle);
```

Explication

- **w** efface le fichier avant d'écrire.
- **fwrite()** écrit la chaîne "Bonjour !".

Manipulation de fichier : Ecrire dans un fichier

Exemple1: Création d'un nouveau fichier avec le mode « w »

```
$idFile = fopen("test.txt", "w");
fwrite($idFile, "Bonjour c'est le " . date("d/m/Y H:i:s"));
fclose($idFile);
```

Exemple2: Modification d'un fichier avec le mode « a »

```
if (file_exists("test.txt")) {
$idFile = fopen("test.txt", "a");
fwrite($idFile, "\nNouvelle ligne");
fclose($idFile);
}
```

//A chaque exécution de ce code, la chaine « Nouvelle ligne » est ajoutée à la fin de ce fichier.

Manipulation de fichier : Ecrire dans un fichier

file_put_contents()

Écrit une chaîne dans un fichier sans fopen().

Syntaxe

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

Exemple

```
file_put_contents("test.txt", "Bonjour !");
//Crée le fichier test.txt s'il n'existe pas. Écrase le contenu s'il existe déjà.

file_put_contents("exemple.txt", "Nouvelle ligne.\n", FILE_APPEND);
// Avec FILE_APPEND, le contenu est ajouté à la fin du fichier sans l'écraser.
```

Manipulation de fichier : Chargement

Un formulaire de téléchargement de fichiers peut être construit en créant un formulaire spécifique comme ceci :

```
            <!-- le type d'encodage des données, enctype, doit être spécifié comme ce qui suit -->
<form action="" method="post" enctype="multipart/form-data">
                    <!-- max_file_size doit précéder le champ input de type file -->
            <input type="hidden" name="max_file_size" value="30000" />
<!-- le nom de l'élément input détermine le nom dans le tableau $_files -->
        choisissez un fichier <input type="file" name="myfile" />
        <br>
        <input type="submit" value="envoi" />
 </form>
```

Manipulation de fichier : Chargement

Après avoir soumettre le formulaire, la variable globale $_FILES contiendra toutes les informations sur le fichier téléchargé.

`$_FILES["myFile"]["name"]`	Le nom original du fichier, tel que sur la machine du client web.
`$_FILES["myFile"]["type"]`	Le type MIME du fichier, si le navigateur a fourni cette information. Par exemple, cela pourra être "image/gif". Ce type mime n'est cependant pas vérifié du côté de PHP et, donc, ne prend pas sa valeur pour se synchroniser.
`$_FILES["myFile"]["size"]`	La taille, en octets, du fichier téléchargé.
`$_FILES["myFile"]["tmp_name"]`	Le nom temporaire du fichier qui sera chargé sur la machine serveur.
`$_FILES["myFile"]["error"]`	Le code d'erreur associé au téléchargement de fichier.

Manipulation de fichier : Chargement

move_uploaded_file(string $source, string $destination): bool

→ Cette fonction **déplace** le fichier du répertoire temporaire vers le dossier du script (ou autre chemin de ton choix).

```php
 <?php
if (isset($_FILES['myFile’']))
 {
//Enregistrement et renommage du fichier
       $nom = "uploaded_" . $_FILES["myFile"]["name"];
       $result = move_uploaded_file($_FILES["myFile"]["tmp_name"], $nom);
          if ($result == TRUE) {
            echo "<b>Vous avez bien transféré le fichier</b><br>";
            echo "Le nom du fichier est : ", $_FILES["myFile"]["name"], "<br>";
            echo "Votre fichier a une taille de ", $_FILES["myFile"]["size"],"<br>";
          } else {
            echo "<hr /> Erreur de transfert n°", $_FILES["myFile"]["error"];
          }
}
 ?>
```