

PHP : Les fichiers

L2 MPCIE - UE Développement Web

David Lesaint
david.lesaint@univ-angers.fr



Janvier 2019

Gestion des fichiers

Notion de flux

Flux (stream)

- Une ressource (type `resource`) accessible linéairement en lecture ou en écriture : fichier, ressource Web, données de BDD, ...
- Identifié avec la syntaxe `scheme://target`
 - `file://*` : accès aux fichiers locaux
 - `http://*` : accès HTTP aux URL
 - `ftp://*` : accès FTP aux URL
 - `zlib://*` : compression de données
 - `phar://*` : archive PHP
 - `ssh2://*` : SSH 2
 - ...

Gestionnaire de protocole (wrapper)

Code PHP pour gérer certains protocoles/encodages liés à un flux.

- eg. le wrapper HTTP convertit une URL en requête HTTP.

Contexte de flux

- Un ensemble de paramètres et d'options spécifiques à un wrapper pour contrôler le comportement d'un flux.
- Se crée avec la fonction `resource stream_context_create([array $options [, $params]])`.
- Se passe ensuite en argument aux fonctions de création et d'accès aux flux : `fopen`, `file_get_contents`, `file_put_contents`, ...

Exemple : accès HTTP

Récupération d'une ressource Web par son URL (file_get_contents.php)

```
1 // création des options du contexte de flux
2 $opts = array (
3     'http'=>array (
4         'method'=>"GET",
5         'header'=>"Accept-language: en\r\n" .
6                 "Cookie: foo=bar\r\n"
7     )
8 );
9 // création du contexte de flux
10 $context = stream_context_create($opts);
11 // accès au flux "http://www.univ-angers.fr"
12 // avec envoi par le wrapper PHP pour HTTP
13 // d'une requête HTTP/1.1
14 // contenant les entêtes HTTP ci-dessus
15 $file = file_get_contents('http://www.univ-angers.fr/', false, $context);
16 echo $file;
```

Gestion des fichiers

Lecture et écriture de fichiers

A l'aide de

- Fonctions : `fopen`, `fread`, `fwrite`, `fclose`, `file_get_contents`, `file_put_contents`, ...
- Deux classes SPL : `SPLFileInfo` et `SPLFileObject`.

Différentes modalités d'accès

- Lecture seule, écriture seule, ou les deux.
- Ecrasement de fichier ou ajout de données par écriture.
- Lecture intégrale, ligne par ligne, caractère par caractère.
- Lecture automatique de données formatées.
- Positionnement de pointeur de lecture/écriture.
- Verrouillage d'accès.
- Copie, renommage, effacement, méta-données.

Lecture de fichier avec `file_get_contents`

```
string file_get_contents(string $f [,...])
```

Lit le contenu d'un fichier en totalité et le retourne dans une chaîne.

- `$f` : nom du fichier.
- Drapeaux :
 - recherche ou non du fichier dans le `$PATH`.
 - contexte de flux valide (`NULL` pour les fichiers).
 - positionnement du pointeur de lecture.
 - nombre d'octets à lire ou bien lecture intégrale.

Retourne `FALSE` en cas d'erreur.

Emet un `E_WARNING` si fichier introuvable.

Lecture de fichier avec `file_get_contents`

Lecture d'un fichier local (`file_get_contents1.php`)

```
1 $filename=__DIR__.'/html.txt';  
2 $str=file_get_contents($filename) ;  
3 if ($str===false) {  
4     throw Exception('could not read') ;  
5 } else {  
6     echo $str ;  
7 }
```

Lecture d'une page HTML (`file_get_contents2.php`)

```
1 $file=file_get_contents("http://www.univ-angers.fr") ;  
2 echo $file ;
```


Écriture de fichier avec `file_put_contents`

```
file_put_contents(string $f, mixed $d [,...])
```

Écrit dans le fichier spécifié par `$f` la donnée `d` :

- `$d` : chaîne, tableau ou ressource de flux.
- Drapeaux :
 - ou-logique entre constantes : `FILE_USE_INCLUDE_PATH` (recherche dans le `$PATH`), `FILE_APPEND` (concaténation vs. écrasement), `LOCK_EX` (verrou exclusif).
 - contexte de flux (`NULL` pour les fichiers)

Crée le fichier s'il n'existe pas, sinon l'écrase ou y ajoute les données selon l'option `FILE_APPEND`.

Retourne le nombre d'octets écrits ou `FALSE` en cas d'erreur.

Ecriture de fichier avec `file_put_contents`

Ecriture dans un fichier local (`file_put_contents.php`)

```
1 $fname=__DIR__.' /file_put_contents1.txt';  
2 $tab=array('odyssée', 'de', 'l\'espace', 2001);  
3 file_put_contents($fname,implode(' ', $tab). "\n");
```

Utilisation de `fopen`, `fwrite`, `fclose`

```
resource fopen(string $f, string $mode [,...])
```

- Ouvre un fichier ou une URL spécifiée par `$f`.
- Drapeaux :
 - `$mode` : lecture seule `"r"`, lecture et écriture `"r+"`, écriture seule avec écrasement `"w"`, lecture et écriture avec écrasement `"w+"`, écriture seule par ajout `"a"`, lecture et écriture par ajout `"a+"`.
 - recherche dans le `$PATH`.
 - contexte de flux valide (`NULL` pour les fichiers).

```
int fwrite(resource $h, string $s [,int $n])
```

- Écrit le contenu de `$s` (maximum `$n` octets) dans le fichier identifié par `$h`.

```
bool fclose(resource $h)
```

- Ferme le fichier identifié par `$h`.

Utilisation de fopen, fwrite, fclose

Exemple de compression de fichier (fopen.php)

```
1  /* création d'un fichier compressé (binaire)
2  * Le fichier peut être décompressé
3  * et lu avec le wrapper compress.zlib stream
4  * ou en ligne de commande 'gzip -d foo-bar.txt.gz'
5  */
6  $fp = fopen("compress.zlib://foo-bar.txt.gz", "wb");
7  if (!$fp) die("Unable to create file.");
8  fwrite($fp, "This is a test.\n");
9  fclose($fp);
```

Autres fonctions

Nom	Description
flock	dé-/verrouillage d'accès.
fgetc	lecture d'un caractère à la fois.
fgets	lecture d'une ligne à la fois (délimiteur <code>\n</code>).
fread	lecture de blocs de taille prédéfinie.
fseek	positionnement du pointeur.
rewind	positionner le pointeur en début de fichier.
ftell	position du pointeur.
fgetcsv	lecture d'une ligne à la fois et conversion sous forme de tableau de mots basée sur séparateur prédéfini.
readfile	lecture intégrale et affichage sur la sortie standard.

Autres fonctions

Nom	Description
<code>copy</code>	copie.
<code>rename</code>	renommage.
<code>unlink</code>	suppression.
<code>file_exists</code>	existence.
<code>filesize</code>	taille.
<code>filetype</code>	de type fichier ou répertoire.
<code>is_file</code>	vérification.
<code>is_readable</code>	accessibilité en lecture.
<code>is_writable</code>	accessibilité en écriture.
<code>is_uploaded_file</code>	test de téléversement.
<code>fileatime</code>	dernier accès.
<code>filemtime</code>	dernière modification.
<code>filectime</code>	dernière modification des permissions.
<code>realpath</code>	chemin d'accès.
<code>basename</code>	extraction du nom à partir d'un chemin.