

Interagir avec le serveur FTP

Par Despirithium



OPENCLASSROOMS

www.openclassrooms.com

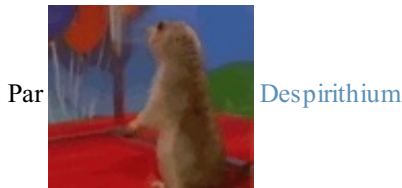
*Licence Creative Commons 6 2.0
Dernière mise à jour le 21/09/2010*

Sommaire

Sommaire	2
Interagir avec le serveur FTP	3
Ouvrir et fermer la connexion	3
Un petit point sur le port	3
Ouvrir une connexion SSL	4
S'identifier	5
Fermer la connexion	5
Lister, télécharger et envoyer des fichiers	5
Lister les fichiers sur le serveur	5
Télécharger un fichier	6
Envoyer un fichier	7
Quelques fonctionnalités intéressantes	8
Créer un dossier : ftp_mkdir	8
Supprimer un dossier : ftp_rmdir	8
Se déplacer à travers les dossiers : ftp_chdir	9
Situer où l'on est : ftp_pwd	9
Exécuter une commande FTP : ftp_exec	9
Modifier les droits : ftp_chmod	10
Partager	11



Interagir avec le serveur FTP



Par

Despirithium

Mise à jour : 21/09/2010

Difficulté : Facile

Durée d'étude : 45 minutes



Bonjour à tous. 😊

Dans ce cours, vous allez apprendre à utiliser les fonctions FTP afin d'interagir avec un serveur FTP. Vous pourrez vous y connecter, envoyer des données (**Upload**) et en recevoir (**Download**), créer des dossiers, en supprimer et bien d'autres choses encore.



Que signifie « FTP » ?
À quoi ça sert ?

« FTP » signifie *File Transfer Protocol* (protocole de transfert de fichiers). Il sert à faire transiter des fichiers entre **un serveur et un client**. C'est un protocole (parmi tant d'autres) très utilisé qui est, en général, associé à un port : le port 21. Ça vous rappelle quelque chose, hein ? 🤔

En effet, lorsque vous utilisez votre client FTP, il y a en général un champ « Port » où vous entrez « 21 » la plupart du temps. Il s'agit du port auquel le protocole FTP va se connecter.

Il n'est donc pas utilisable seulement avec des logiciels. PHP propose notamment de puissants outils afin d'utiliser le protocole FTP en toute simplicité.

Prêts ? C'est parti. 😊

Sommaire du tutoriel :



- Ouvrir et fermer la connexion
- Lister, télécharger et envoyer des fichiers
- Quelques fonctionnalités intéressantes

Ouvrir et fermer la connexion

Pour commencer, il vous faut ouvrir une connexion avec votre serveur FTP. La fonction `ftp_connect` retourne une ressource FTP qui vous servira pour le maniement de vos fichiers. Elle s'utilise ainsi :

Code : PHP

```
<?php
$ftp = ftp_connect("hôte", "port");
?>
```

Un petit point sur le port

Si vous n'avez jamais compris ce qu'était le port, alors il est grand temps de faire un petit *speech* à ce sujet. Lorsque l'on communique sur le réseau par le biais d'un protocole, l'ordinateur laisse plusieurs « passages » par lesquels un client et un serveur peuvent communiquer : c'est ce que l'on appelle les **ports**. Grosso modo, en vous connectant à un port, vous assurez un autre moyen de communication avec le serveur.

Un ordinateur laisse 65535 ports, dont les 1024 premiers sont souvent utilisés par les programmes « par défaut » ou « communs ». Par exemple, lorsque vous utilisez votre navigateur web, vous vous connectez au port 80 du serveur web afin d'afficher votre page web. Eh bien c'est pareil pour le FTP !

Bref, étant donné que la plupart des serveurs FTP écoutent le port 21, on n'a qu'à emprunter ce chemin. 😊

Après ce résumé sur les ports, revenons à nos moutons.

Si je veux me connecter à un serveur FTP situé chez Free, étant donné que leur domaine FTP est `ftpperso.free.fr`, le code sera le suivant :

Code : PHP

```
<?php
$ftp = ftp_connect("ftpperso.free.fr", 21);
?>
```

Vous pouvez également utiliser `or exit()` juste après l'utilisation de cette fonction :

Code : PHP

```
<?php
$ftp = ftp_connect("ftpperso.free.fr", 21) or exit('Erreur :
connexion au serveur FTP impossible.');
```

Vous pouvez omettre le second paramètre (qui est le port), mais par habitude et pour plus de lisibilité, je le renseigne. Cette fonction retourne `false` si la connexion n'a pas été possible, mais une erreur est quand même écrite dans ce cas.

Ouvrir une connexion SSL



Hein ? Qu'est-ce que SSL ?

Citation : Sécurité Info

SSL (*Secure Socket Layer*) est un protocole de sécurisation des échanges, développé par Netscape. Il a été conçu pour assurer la sécurité des transactions sur Internet (notamment entre un client et un serveur) et il est intégré depuis 1994 dans les navigateurs.

Cela signifie donc qu'il permet de sécuriser les échanges entre le client et le serveur.

Seulement, lorsque l'on utilise une connexion SSL, on ne parle plus proprement de FTP mais de **FTPS : *File Transfer Protocol over SSL*** (tout est dit). Certes les manipulations restent les mêmes, mais il est bon de savoir dans quoi on marche. 😊

Bref, pour se connecter, rien de plus simple, il suffit de procéder de la même manière que pour `ftp_connect`, seul le nom de la fonction est différente :

Code : PHP

```
<?php
$ftp = ftp_ssl_connect("hôte", "port");
?>
```

Il n'y a rien d'autre à faire, tous les traitements se font automatiquement. 😊

Voilà, vous avez ouvert une connexion entre le serveur HTTP et le serveur FTP. En effet, vous ne vous connectez pas au serveur

FTP mais vous demandez au serveur HTTP de le faire (d'ailleurs, il peut se connecter à plusieurs serveurs). Schématiquement, une utilisation des fonctions FTP sur une page web donnerait :
Utilisateur ? Connexion au serveur HTTP (exécution de la page) ? Demande de connexion au serveur FTP ? **Interaction** avec le serveur FTP ? Déconnexion du serveur FTP.

S'identifier

Maintenant que vous êtes connecté au serveur, il faut vous connecter à un compte FTP. En effet, lorsque, par exemple, vous utilisez un client FTP (comme FileZilla), vous entrez également un identifiant et un mot de passe. Pour cela, il faut utiliser la fonction `ftp_login` qui prend trois paramètres :

- la ressource FTP ;
- l'identifiant ;
- le mot de passe.

Code : PHP

```
<?php
ftp_login($ftp, "login", "password");
?>
```

La fonction renvoie true si la connexion a réussi, false en cas d'échec.



Il faut obligatoirement être identifié pour pouvoir interagir avec le serveur FTP, certains utilisateurs ayant des droits plus restreints que d'autres.

Fermer la connexion

Une fois vos manipulations finies, vous pouvez fermer votre connexion avec la fonction `ftp_close`, qui prend en paramètre la ressource FTP utilisée lors de la connexion.

Code : PHP

```
<?php
ftp_close($ftp);
?>
```

Justement, et si on commençait à le manipuler, ce serveur FTP ?

Lister, télécharger et envoyer des fichiers

Nous allons voir trois fonctions de base mais qui vous seront très utiles (voire indispensables) si vous voulez créer un WebFTP (comme chez les hébergeurs).

Lister les fichiers sur le serveur

Vous pouvez lister les fichiers (et dossiers) d'un dossier en utilisant la fonction `ftp_nlist` qui prend en paramètres la ressource FTP et le dossier dont il faut lister les fichiers.

Code : PHP

```
<?php
$fichiers = ftp_nlist($ftp, "."); // Le point signifie le dossier
```

```
actuel  
?>
```

Il retourne un tableau avec le nom de tous les fichiers. Voici donc un exemple de code complet pour lister les fichiers sur votre serveur FTP :

Code : PHP

```
<?php  
if (($ftp = ftp_connect("ftpperso.free.fr", 21)) == false)  
{  
    echo 'Erreur de connexion...';  
}  
  
if (!ftp_login($ftp, "login", "password"))  
{  
    echo 'L\'identification a échoué...';  
}  
  
$liste_fichiers = ftp_nlist($ftp, '.');  
  
foreach ($liste_fichiers as $fichier)  
{  
    echo $fichier. '<br/>';  
}  
?>
```

Un petit problème d'envergure

Il arrive parfois que votre connexion sur le serveur FTP passe par un **Firewall** ou un **NAT** (chez le client, soit vous, évidemment). Dans ce cas, il faut activer le mode passif, car sinon vous aurez des problèmes lors de vos manipulations avec le serveur FTP (par exemple, pour la fonction `ftp_nlist`, puisqu'on est en plein dedans).

Pour activer le mode passif, il suffit simplement d'utiliser la fonction `ftp_pasv`. Elle retourne « true » si le changement de mode a bien été effectué.

Code : PHP

```
<?php  
if (ftp_pasv($ftp, true)) // La valeur « true » signifie qu'on  
    l'active, false qu'on le désactive  
{  
    // Alors on peut commencer nos traitements  
}  
else  
{  
    // Dans ce cas, pour une raison ou une autre, le changement de  
    mode a échoué.  
}  
?>
```

Prenez toujours le soin de passer en mode passif lorsque votre code est bon, mais que vous n'arrivez tout de même pas à interagir avec votre serveur FTP. Ça peut être très rageant de planter sur un bout de code toute une après-midi alors que l'erreur vient simplement du réseau lui-même. 😊

Télécharger un fichier

Ah, enfin ! Comment faire pour télécharger un fichier ? Il faut utiliser la fonction `ftp_get` .

Code : PHP

```
<?php
ftp_get($ftp, "dossier où envoyer", "fichier à télécharger",
"mode");
?>
```

Expliquons tout ça :

- le premier paramètre est la ressource FTP créée plus tôt ;
- le second est le dossier où le fichier sera envoyé sur **l'ordinateur du client** ;
- le troisième paramètre prend le nom du fichier à télécharger sur le serveur ;
- le dernier paramètre est le mode de téléchargement : binaire ou ASCII. Il faut mettre « `FTP_BINARY` » ou « `FTP_ASCII` ». En général, on met « `FTP_BINARY` ».

Voici un exemple où les fonctions FTP deviennent réellement utiles :

Code : PHP

```
<?php
$ftp = ftp_connect("ftpperso.free.fr", 21);
ftp_login($ftp, "blackis", "*****");

$liste_fichiers = ftp_nlist($ftp, '.');

foreach($liste_fichiers as $fichier)
{
    echo '<a href="?filename=' . $fichier . '">' . $fichier . '</a><br/>';
}

if(isset($_GET['filename']))
{
    $nom_fichier = $_GET['filename'];
    ftp_get($ftp, "C:/", $nom_fichier, FTP_BINARY);
}
?>
```

C'est simple. Ce code liste les fichiers existant sur la racine avec `ftp_nlist` , et lors du clic, on télécharge ledit fichier dans le dossier lien local avec `ftp_get` .

Envoyer un fichier

La méthode `ftp_put` permet d'envoyer un fichier sur le serveur, et s'utilise de cette manière :

Code : PHP

```
<?php
ftp_put($ftp, "dossier où le fichier sera envoyé (serveur)",
"Fichier local", type);
?>
```

Ici aussi, *type* est soit « `FTP_ASCII` », soit « `FTP_BINARY` », et on utilise généralement « `FTP_BINARY` ». La fonction renverra `true` si le fichier a bien été envoyé ou `false` dans le cas contraire.

Ainsi, si l'on fait :

Code : PHP

```
<?php
ftp_put($ftp, "images/superimage2.jpg", "C:/superimage.jpg",
FTP_BINARY);
?>
```

Le fichier superimage.jpg sera envoyé dans le dossier images du serveur (en partant de la racine), sous le nom superimage2.jpg. (En effet, il faut également renseigner le nom que portera le fichier envoyé sur le serveur FTP, c'est donc tout bénéfice).

Quelques fonctionnalités intéressantes

Créer un dossier : ftp_mkdir

Cette fonction est très connue dans la plupart des langages. Elle permet de créer un dossier (mk : *make* et dir : *directory*, soit « créer un dossier ») sur le serveur.

Elle s'utilise ainsi :

Code : PHP

```
<?php
$mon_dossier = ftp_mkdir($ftp, "superdossier");
?>
```

Ainsi, ce code créera un dossier à la racine (on n'a pas précisé de dossier parent) portant le nom « superdossier ».

Par ailleurs, cette fonction retourne le nom du dossier créé ou false en cas d'échec.

Si l'on veut créer un dossier dans un autre dossier :

Code : PHP

```
<?php
$mon_dossier = ftp_mkdir($ftp, "site/image/design/superdossier");
?>
```

Et voilà, le dossier superdossier a été créé à l'emplacement site/image/design.

Supprimer un dossier : ftp_rmdir

Également connue, la fonction ftp_rmdir (*remove directory*, soit « supprimer un dossier ») supprime un dossier sur le serveur FTP.

Elle prend comme paramètres la ressource FTP et le nom du dossier à supprimer, et retourne true si le dossier a bien été supprimé (false en cas d'échec).

Code : PHP

```
<?php
if(ftp_rmdir($ftp, "superdossier"))
{
    echo 'Le dossier "superdossier" a bien été supprimé !';
}
else
{
    echo 'Suppression du dossier "superdossier" impossible.';
}
```



```
?>
```

Se déplacer à travers les dossiers : ftp_chdir

La fonction `ftp_chdir` (soit *change directory* : « changer de dossier ») vous déplace dans un autre dossier. Elle prend en paramètres la ressource FTP et le dossier dans lequel on souhaite aller, et retourne `true` si l'on a bien pu se déplacer, `false` dans le cas contraire.

Code : PHP

```
<?php
if(ftp_chdir($ftp, "site"))
{
    echo 'Déplacement dans le dossier "site" réussi !';
}
else
{
    echo 'Déplacement dans le dossier "site" <b>échoué</b> !';
}
?>
```

Situer où l'on est : ftp_pwd

La fonction `ftp_pwd` (*pwd* : *print working directory*, soit « afficher le dossier dans lequel on travaille ») retourne (et n'affiche pas comme on pourrait le croire) le nom du dossier dans lequel on se situe. Elle prend en paramètre la ressource FTP seulement. En cas d'erreur, elle retourne `false`.

Code : PHP

```
<?php
$pwd = ftp_pwd($ftp);

if(!$pwd)
{
    echo 'Erreur !';
}
else echo 'Dossier actuel : ' . $pwd . '.';
?>
```

Exécuter une commande FTP : ftp_exec

La fonction `ftp_exec` exécute (🤖) la commande envoyée en second paramètre (le premier étant toujours la ressource FTP) et renvoie `true` (ainsi que la réponse 200 pour dire que tout a bien été fait correctement dans le meilleur des mondes) si la commande a réussi, `false` dans le cas contraire.

Code : PHP

```
<?php
if(ftp_exec($ftp, "rmdir superdossier"))
{
    echo 'La commande a réussi !';
}
else
{
    echo 'La commande a échoué !';
}
?>
```

Voici une [page](#) qui liste toutes les actions FTP possibles. C'est plus poussé que les fonctions mais un petit peu plus compliqué.



Modifier les droits : ftp_chmod



Qu'est-ce que le chmod ?

Chmod signifie *change mode* et sert à modifier les droits d'un fichier. En effet, chaque fichier transitant sur le web a trois droits différents :

- les droits du propriétaire du fichier ;
- les droits du groupe auquel le fichier appartient ;
- les droits des autres utilisateurs.

En général, le propriétaire du fichier et le groupe auquel appartient le fichier ont tous les droits, du fait qu'ils sont censés faire « partie de l'administration du serveur ». Quant aux autres, ils ont en général seulement le droit de lire le fichier et de l'exécuter.



Pourquoi les autres n'ont-ils que le droit de lire ?


Prenons un exemple. Vous avez fait un super script en PHP qui permet de gérer le livre d'or. Si n'importe qui peut lire et y écrire :


- on peut récupérer vos mots de passe d'accès à la base de données, connaître l'architecture de votre site ainsi que le nom des tables de votre base de données ;
- on peut vider votre base de données, récupérer toutes les entrées ou en modifier certaines (pour tricher dans un jeu par exemple) ;
- on peut y écrire et y mettre du code malveillant, comme récupérer le pseudo du membre connecté et l'insérer dans une base de données.

En gros, ça peut mettre votre site en danger, c'est pour cela qu'il faut bien faire attention à ne pas laisser les utilisateurs faire ce qu'ils veulent avec vos fichiers.

On distingue donc trois « valeurs » possibles :

- écrire dans le fichier (w, pour *write*) ;
- lire dans le fichier (r, pour *read*) ;
- exécuter le fichier (x, pour *execute*).

Bref, laisser les utilisateurs lire et écrire dans le fichier, c'est vraiment dangereux. Il ne faut jamais faire confiance aux utilisateurs, ils sont imprévisibles, se terrent dans l'ombre, attendant le moment propice pour défaire tout votre site. 

Mais n'ayez pas peur pour autant, il suffit juste de mettre les droits des utilisateurs à l'exécution seulement. 

Chacun a une valeur :

- lire vaut 4 ;
- écrire vaut 2 ;
- exécuter vaut 1.

Ce qui signifie que si l'on veut pouvoir lire et écrire dans un fichier, la valeur est 6.

Maintenant, les droits sont répartis ainsi : Propriétaire - Groupe - Utilisateurs.

Par habitude, je mets au Groupe et au Propriétaire les mêmes droits. Donc, si je veux que le propriétaire et le groupe puissent écrire, lire et exécuter, et que les utilisateurs puissent lire et exécuter, il faut donc faire :

Secret (cliquez pour afficher)

775

Si vous avez trouvé ça, vous avez tout compris. 😊

Donc, la fonction `ftp_chmod` prend en paramètres la ressource FTP, le mode et le fichier affecté par la modification. Elle retourne `true` si le fichier a bien été affecté, `false` dans le cas contraire.

Code : PHP

```
<?php
if(ftp_chmod($ftp, 0775, "fichier"))
{
    echo 'Les droits du fichier ont bien été modifiés !';
}
else echo 'Les droits du fichier n\'ont pas pu être modifiés !';
?>
```



Pourquoi ce 0 devant le « 775 » ?

Ce petit 0 signifie que l'on utilise la notation octale (base 8). Si on ne l'avait pas mis, la fonction l'aurait interprété comme un nombre décimal (base 10).

Voilà, si vous avez bien mis un fichier valide, tout devrait fonctionner.

Voilà, déjà la fin du tutoriel. 😊

Bien heureusement, je n'ai listé que la moitié (et encore) des fonctions FTP existantes avec PHP.

Voici quelques petites pistes :

- vous pouvez **renommer** un fichier avec une fonction portant le même nom (en anglais, *of course*) ;
- bien évidemment, vous pouvez **supprimer** un fichier !
- vous pouvez également connaître la **taille** d'un fichier ;
- vous pouvez activer / désactiver le **mode passif** (faites quelques recherches si ça vous intéresse !) ;
- vous pouvez retourner au dossier parent avec la fonction `ftp_cdup` !
- il y a une fonction qui vous permet de **lister** tous vos fichiers plus facilement, et avec plus d'informations (la date, le propriétaire, les droits, la taille en octets...) ;
- vous pouvez activer / désactiver des **options** (comme par exemple le « **délai de connexion** »).

Bonne chance pour la suite. 😊

Merci à [ms_fragger](#) pour avoir relevé une erreur que j'avais totalement loupée. 😊

Merci également à [sebdia](#) pour la petite astuce concernant le mode passif. 😊

Partager



Ce tutoriel a été corrigé par les [zCorrecteurs](#).