FTP

Introduction

The functions in this extension implement client access to file servers speaking the File Transfer Protocol (FTP) as defined in » http://www.faqs.org/rfcs/rfc959. This extension is meant for detailed access to an FTP server providing a wide range of control to the executing script. If you only wish to read from or write to a file on an FTP server, consider using the *ftp://* wrapper with the filesystem functions which provide a simpler and more intuitive interface.

Installing/Configuring

Requirements

No external libraries are needed to build this extension.

Installation

In order to use FTP functions with your PHP configuration, you should add the *--enable-ftp* option when installing PHP.

The Windows version of PHP has built-in support for this extension. You do not need to load any additional extensions in order to use these functions.

Runtime Configuration

This extension has no configuration directives defined in *php.ini*.

Resource Types

This extension uses one resource type, which is the link identifier of the FTP connection, returned by ftp_connect() or ftp_ssl_connect().

Predefined Constants

The constants below are defined by this extension, and will only be available when the extension has either been compiled into PHP or dynamically loaded at runtime.

FTP_ASCII (integer)

FTP_TEXT (integer)

FTP_BINARY (integer)

FTP_IMAGE (integer)

FTP_TIMEOUT_SEC (integer)
See ftp_set_option() for information.

The following constants were introduced in PHP 4.3.0.

FTP_AUTOSEEK (integer)
See ftp_set_option() for information.

FTP_AUTORESUME (integer)
Automatically determine resume position and start position for GET and PUT requests (only works if FTP_AUTOSEEK is enabled)

FTP_FAILED (integer)
Asynchronous transfer has failed

FTP_FINISHED (integer)
Asynchronous transfer has finished

FTP_MOREDATA (integer)
Asynchronous transfer is still active

Examples

Example #1 - FTP example

```
<?php
// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// check connection
if ((!$conn_id) || (!$login_result)) {
        echo "FTP connection has failed!";
        echo "Attempted to connect to $ftp_server for user $ftp_user_name";
        exit;
   } else {
        echo "Connected to $ftp_server, for user $ftp_user_name";
   }

// upload the file
$upload = ftp_put($conn_id, $destination_file, $source_file, FTP_BINARY);

// check upload status
if (!$upload) {
        echo "FTP upload has failed!";
   } else {
        echo "Uploaded $source_file to $ftp_server as $destination_file";
   }

// close the FTP stream
ftp_close($conn_id);
?>
```

FTP Functions

ftp_alloc

ftp_alloc -- Allocates space for a file to be uploaded

Description

bool **ftp_alloc** (resource `$ftp_stream`, int `$filesize` [, string `&$result`])

Sends an *ALLO* command to the remote FTP server to allocate space for a file to be uploaded.

Note
Many FTP servers do not support this command. These servers may return a failure code (**FALSE**) indicating the command is not supported or a success code (**TRUE**) to indicate that pre-allocation is not necessary and the client should continue as though the operation were successful. Because of this, it may be best to reserve this function for servers which explicitly require preallocation.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`filezise`
The number of bytes to allocate.

`return`
A textual representation of the servers response will be returned by reference in *`result`* if a variable is provided.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #2 - ftp_alloc() **example**

```
<?php
```

```
$file = "/home/user/myfile";

/* connect to the server */
$conn_id = ftp_connect('ftp.example.com');
$login_result = ftp_login($conn_id, 'anonymous', 'user@example.com');

if (ftp_alloc($conn_id, filesize($file), $result)) {
 echo "Space successfully allocated on server.  Sending $file.\n";
 ftp_put($conn_id, '/incomming/myfile', $file, FTP_BINARY);
} else {
 echo "Unable to allocate space on server.  Server said: $result\n";
}

ftp_close($conn_id);

?>
```

See Also

- ftp_put()
- ftp_fput()

ftp_cdup

ftp_cdup -- Changes to the parent directory

Description

bool **ftp_cdup** (resource `$ftp_stream`)

Changes to the parent directory.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #3 - ftp_cdup() example

```
<?php
// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// change the current directory to html
ftp_chdir($conn_id, 'html');

echo ftp_pwd($conn_id); // /html

// return to the parent directory
if (ftp_cdup($conn_id)) {
 echo "cdup successful\n";
} else {
 echo "cdup not successful\n";
}

echo ftp_pwd($conn_id); // /

ftp_close($conn_id);
?>
```

See Also

- ftp_chdir()
- ftp_pwd()

ftp_chdir

ftp_chdir -- Changes the current directory on a FTP server

Description

bool **ftp_chdir** (resource `$ftp_stream`, string `$directory`)

Changes the current directory to the specified one.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`directory`
The target directory.

Return Values

Returns **TRUE** on success or **FALSE** on failure. If changing directory fails, PHP will also throw a warning.

Examples

Example #4 - ftp_chdir() example

```
<?php

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// check connection
if ((!$conn_id) || (!$login_result)) {
   die("FTP connection has failed !");
}

echo "Current directory: " . ftp_pwd($conn_id) . "\n";

// try to change the directory to somedir
if (ftp_chdir($conn_id, "somedir")) {
   echo "Current directory is now: " . ftp_pwd($conn_id) . "\n";
} else {
   echo "Couldn't change directory\n";
```

```
}

// close the connection
ftp_close($conn_id);
?>
```

See Also

- ftp_cdup()
- ftp_pwd()

ftp_chmod

ftp_chmod -- Set permissions on a file via FTP

Description

int **ftp_chmod** (resource `$ftp_stream`, int `$mode`, string `$filename`)

Sets the permissions on the specified remote file to *`mode`*.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`mode`
The new permissions, given as an *octal* value.

`filename`
The remote file.

Return Values

Returns the new file permissions on success or **FALSE** on error.

Examples

Example #5 - ftp_chmod() **example**

```
<?php
$file = 'public_html/index.php';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to chmod $file to 644
if (ftp_chmod($conn_id, 0644, $file) !== false) {
echo "$file chmoded successfully to 644\n";
} else {
echo "could not chmod $file\n";
}

// close the connection
ftp_close($conn_id);
```

```
?>
```

See Also

- chmod()

ftp_close

ftp_close -- Closes an FTP connection

Description

bool **ftp_close** (resource `$ftp_stream`)

ftp_close() closes the given link identifier and releases the resource.

Note
After calling this function, you can no longer use the FTP connection and must create a new one with ftp_connect().

Parameters

`ftp_stream`
The link identifier of the FTP connection.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #6 - ftp_close() example

```
<?php

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// print the current directory
echo ftp_pwd($conn_id); // /

// close this connection
ftp_close($conn_id);
?>
```

See Also

- ftp_connect()

ftp_connect

ftp_connect -- Opens an FTP connection

Description

resource **ftp_connect** (string `$host` [, int `$port` [, int `$timeout`]])

ftp_connect() opens an FTP connection to the specified *`host`*.

Parameters

`host`
: The FTP server address. This parameter shouldn't have any trailing slashes and shouldn't be prefixed with *ftp://*.

`port`
: This parameter specifies an alternate port to connect to. If it is omitted or set to zero, then the default FTP port, 21, will be used.

`timeout`
: This parameter specifies the timeout for all subsequent network operations. If omitted, the default value is 90 seconds. The timeout can be changed and queried at any time with ftp_set_option() and ftp_get_option().

Return Values

Returns a FTP stream on success or **FALSE** on error.

Examples

Example #7 - ftp_connect() example

```
<?php

$ftp_server = "ftp.example.com";

// set up a connection or die
$conn_id = ftp_connect($ftp_server) or die("Couldn't connect to
$ftp_server");

?>
```

ChangeLog

Version	Description
4.2.0	*timeout* was added.

See Also

- ftp_close()
- ftp_ssl_connect()

ftp_delete

ftp_delete -- Deletes a file on the FTP server

Description

bool **ftp_delete** (resource `$ftp_stream`, string `$path`)

ftp_delete() deletes the file specified by *`path`* from the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`path`
The file to delete.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #8 - ftp_delete() example

```
<?php
$file = 'public_html/old.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to delete $file
if (ftp_delete($conn_id, $file)) {
echo "$file deleted successful\n";
} else {
echo "could not delete $file\n";
}

// close the connection
ftp_close($conn_id);
?>
```

ftp_exec

ftp_exec -- Requests execution of a command on the FTP server

Description

bool **ftp_exec** (resource `$ftp_stream`, string `$command`)

Sends a SITE EXEC *`command`* request to the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`command`
The command to execute.

Return Values

Returns **TRUE** if the command was successful (server sent response code: *200*); otherwise returns **FALSE**.

Examples

Example #9 - ftp_exec() example

```
<?php

// variable initialization
$command = 'ls -al >files.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// execute command
if (ftp_exec($conn_id, $command)) {
   echo "$command executed successfully\n";
} else {
   echo "could not execute $command\n";
}

// close the connection
ftp_close($conn_id);
```

```
?>
```

See Also

- ftp_raw()

ftp_fget

ftp_fget -- Downloads a file from the FTP server and saves to an open file

Description

bool **ftp_fget** (resource `$ftp_stream`, resource `$handle`, string `$remote_file`, int `$mode` [, int `$resumepos`])

ftp_fget() retrieves *`remote_file`* from the FTP server, and writes it to the given file pointer.

Parameters

`ftp_stream`
: The link identifier of the FTP connection.

`handle`
: An open file pointer in which we store the data.

`remote_file`
: The remote file path.

`mode`
: The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`resumepos`
: The position in the remote file to start downloading from.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #10 - ftp_fget() example

```
<?php

// path to remote file
$remote_file = 'somefile.txt';
$local_file = 'localfile.txt';

// open some file to write to
$handle = fopen($local_file, 'w');

// set up basic connection
```

```
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to download $remote_file and save it to $handle
if (ftp_fget($conn_id, $handle, $remote_file, FTP_ASCII, 0)) {
echo "successfully written to $local_file\n";
} else {
echo "There was a problem while downloading $remote_file to $local_file\n";
}

// close the connection and the file handler
ftp_close($conn_id);
fclose($handle);
?>
```

ChangeLog

Version	Description
4.3.0	*resumepos* was added.

See Also

- ftp_get()
- ftp_nb_get()
- ftp_nb_fget()

ftp_fput

ftp_fput -- Uploads from an open file to the FTP server

Description

bool **ftp_fput** (resource `$ftp_stream`, string `$remote_file`, resource `$handle`, int `$mode` [, int `$startpos`])

ftp_fput() uploads the data from a file pointer to a remote file on the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The remote file path.

`handle`
An open file pointer on the local file. Reading stops at end of file.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`startpos`

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #11 - ftp_fput() example

```
<?php

// open some file for reading
$file = 'somefile.txt';
$fp = fopen($file, 'r');

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
```

```
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to upload $file
if (ftp_fput($conn_id, $file, $fp, FTP_ASCII)) {
   echo "Successfully uploaded $file\n";
} else {
   echo "There was a problem while uploading $file\n";
}

// close the connection and the file handler
ftp_close($conn_id);
fclose($fp);

?>
```

ChangeLog

Version	Description
4.3.0	*startpos* was added.

See Also

- ftp_put()
- ftp_nb_fput()
- ftp_nb_put()

ftp_get_option

ftp_get_option -- Retrieves various runtime behaviours of the current FTP stream

Description

mixed **ftp_get_option** (resource `$ftp_stream`, int `$option`)

This function returns the value for the requested *`option`* from the specified FTP connection.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`option`
Currently, the following options are supported:

Supported runtime FTP options

FTP_TIMEOUT_SEC	Returns the current timeout used for network related operations.
FTP_AUTOSEEK	Returns **TRUE** if this option is on, **FALSE** otherwise.

Return Values

Returns the value on success or **FALSE** if the given *`option`* is not supported. In the latter case, a warning message is also thrown.

Examples

Example #12 - ftp_get_option() example

```
<?php
// Get the timeout of the given FTP stream
$timeout = ftp_get_option($conn_id, FTP_TIMEOUT_SEC);
?>
```

See Also

- ftp_set_option()

ftp_get

ftp_get -- Downloads a file from the FTP server

Description

bool **ftp_get** (resource `$ftp_stream`, string `$local_file`, string `$remote_file`, int `$mode` [, int `$resumepos`])

ftp_get() retrieves a remote file from the FTP server, and saves it into a local file.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`local_file`
The local file path (will be overwritten if the file already exists).

`remote_file`
The remote file path.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`resumepos`
The position in the remote file to start downloading from.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #13 - ftp_get() example

```
<?php

// define some variables
$local_file = 'local.zip';
$server_file = 'server.zip';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
```

```
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to download $server_file and save to $local_file
if (ftp_get($conn_id, $local_file, $server_file, FTP_BINARY)) {
   echo "Successfully written to $local_file\n";
} else {
   echo "There was a problem\n";
}

// close the connection
ftp_close($conn_id);

?>
```

ChangeLog

Version	Description
4.3.0	*resumepos* was added.

See Also

- ftp_pasv()
- ftp_fget()
- ftp_nb_get()
- ftp_nb_fget()

ftp_login

ftp_login -- Logs in to an FTP connection

Description

bool **ftp_login** (resource `$ftp_stream`, string `$username`, string `$password`)

Logs in to the given FTP stream.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`username`
The username (*USER*).

`password`
The password (*PASS*).

Return Values

Returns **TRUE** on success or **FALSE** on failure. If login fails, PHP will also throw a warning.

Examples

Example #14 - ftp_login() example

```
<?php

$ftp_server = "ftp.example.com";
$ftp_user = "foo";
$ftp_pass = "bar";

// set up a connection or die
$conn_id = ftp_connect($ftp_server) or die("Couldn't connect to
$ftp_server");

// try to login
if (@ftp_login($conn_id, $ftp_user, $ftp_pass)) {
   echo "Connected as $ftp_user@$ftp_server\n";
} else {
   echo "Couldn't connect as $ftp_user\n";
}
```

```
// close the connection
ftp_close($conn_id);
?>
```

ftp_mdtm

ftp_mdtm -- Returns the last modified time of the given file

Description

int **ftp_mdtm** (resource `$ftp_stream`, string `$remote_file`)

ftp_mdtm() gets the last modified time for a remote file.

Note
Not all servers support this feature!

Note
ftp_mdtm() does not work with directories.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The file from which to extract the last modification time.

Return Values

Returns the last modified time as a Unix timestamp on success, or -1 on error.

Examples

Example #15 - ftp_mdtm() example

```
<?php

$file = 'somefile.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
```

```
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

//  get the last modified time
$buff = ftp_mdtm($conn_id, $file);

if ($buff != -1) {
   // somefile.txt was last modified on: March 26 2003 14:16:41.
   echo "$file was last modified on : " . date("F d Y H:i:s.", $buff);
} else {
   echo "Couldn't get mdtime";
}

// close the connection
ftp_close($conn_id);

?>
```

ftp_mkdir

ftp_mkdir -- Creates a directory

Description

string **ftp_mkdir** (resource `$ftp_stream`, string `$directory`)

Creates the specified *`directory`* on the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`directory`
The name of the directory that will be created.

Return Values

Returns the newly created directory name on success or **FALSE** on error.

Examples

Example #16 - ftp_mkdir() example

```
<?php

$dir = 'www';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to create the directory $dir
if (ftp_mkdir($conn_id, $dir)) {
echo "successfully created $dir\n";
} else {
echo "There was a problem while creating $dir\n";
}

// close the connection
ftp_close($conn_id);
?>
```

See Also

- ftp_rmdir()

ftp_nb_continue

ftp_nb_continue -- Continues retrieving/sending a file (non-blocking)

Description

int **ftp_nb_continue** (resource `$ftp_stream`)

Continues retrieving/sending a file non-blocking.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

Return Values

Returns **FTP_FAILED** or **FTP_FINISHED** or **FTP_MOREDATA**.

Examples

Example #17 - ftp_nb_continue() example

```
<?php

// Initate the download
$ret = ftp_nb_get($my_connection, "test", "README", FTP_BINARY);
while ($ret == FTP_MOREDATA) {

  // Continue downloading...
  $ret = ftp_nb_continue($my_connection);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error downloading the file...";
  exit(1);
}
?>
```

ftp_nb_fget

ftp_nb_fget -- Retrieves a file from the FTP server and writes it to an open file (non-blocking)

Description

int **ftp_nb_fget** (resource `$ftp_stream`, resource `$handle`, string `$remote_file`, int `$mode` [, int `$resumepos`])

ftp_nb_fget() retrieves a remote file from the FTP server.

The difference between this function and ftp_fget() is that this function retrieves the file asynchronously, so your program can perform other operations while the file is being downloaded.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`handle`
An open file pointer in which we store the data.

`remote_file`
The remote file path.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`resumepos`

Return Values

Returns **FTP_FAILED** or **FTP_FINISHED** or **FTP_MOREDATA**.

Examples

Example #18 - ftp_nb_fget() example

```
<?php

// open some file for reading
$file = 'index.php';
```

```
$fp = fopen($file, 'w');

$conn_id = ftp_connect($ftp_server);

$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// Initate the download
$ret = ftp_nb_fget($conn_id, $fp, $file, FTP_BINARY);
while ($ret == FTP_MOREDATA) {

  // Do whatever you want
  echo ".";

  // Continue downloading...
  $ret = ftp_nb_continue($conn_id);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error downloading the file...";
  exit(1);
}

// close filepointer
fclose($fp);
?>
```

See Also

- ftp_nb_get()
- ftp_nb_continue()
- ftp_fget()
- ftp_get()

ftp_nb_fput

ftp_nb_fput -- Stores a file from an open file to the FTP server (non-blocking)

Description

int **ftp_nb_fput** (resource `$ftp_stream`, string `$remote_file`, resource `$handle`, int `$mode` [, int `$startpos`])

ftp_nb_fput() uploads the data from a file pointer to a remote file on the FTP server.

The difference between this function and the ftp_fput() is that this function uploads the file asynchronously, so your program can perform other operations while the file is being uploaded.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The remote file path.

`handle`
An open file pointer on the local file. Reading stops at end of file.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`startpos`

Return Values

Returns **FTP_FAILED** or **FTP_FINISHED** or **FTP_MOREDATA**.

Examples

Example #19 - ftp_nb_fput() example

```
<?php

$file = 'index.php';

$fp = fopen($file, 'r');
```

```
$conn_id = ftp_connect($ftp_server);

$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// Initate the upload
$ret = ftp_nb_fput($conn_id, $file, $fp, FTP_BINARY);
while ($ret == FTP_MOREDATA) {

  // Do whatever you want
  echo ".";

  // Continue upload...
  $ret = ftp_nb_continue($conn_id);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error uploading the file...";
  exit(1);
}

fclose($fp);
?>
```

See Also

- ftp_nb_put()
- ftp_nb_continue()
- ftp_put()
- ftp_fput()

ftp_nb_get

ftp_nb_get -- Retrieves a file from the FTP server and writes it to a local file (non-blocking)

Description

int **ftp_nb_get** (resource `$ftp_stream`, string `$local_file`, string `$remote_file`, int `$mode` [, int `$resumepos`])

ftp_nb_get() retrieves a remote file from the FTP server, and saves it into a local file.

The difference between this function and ftp_get() is that this function retrieves the file asynchronously, so your program can perform other operations while the file is being downloaded.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`local_file`
The local file path (will be overwritten if the file already exists).

`remote_file`
The remote file path.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`resumepos`

Return Values

Returns **FTP_FAILED** or **FTP_FINISHED** or **FTP_MOREDATA**.

Examples

Example #20 - ftp_nb_get() example

```
<?php

// Initate the download
$ret = ftp_nb_get($my_connection, "test", "README", FTP_BINARY);
while ($ret == FTP_MOREDATA) {
```

```
  // Do whatever you want
  echo ".";

  // Continue downloading...
  $ret = ftp_nb_continue($my_connection);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error downloading the file...";
  exit(1);
}
?>
```

Example #21 - Resuming a download with ftp_nb_get()

```
<?php

// Initate
$ret = ftp_nb_get($my_connection, "test", "README", FTP_BINARY,
                  filesize("test"));
// OR: $ret = ftp_nb_get($my_connection, "test", "README",
//                       FTP_BINARY, FTP_AUTORESUME);
while ($ret == FTP_MOREDATA) {

  // Do whatever you want
  echo ".";

  // Continue downloading...
  $ret = ftp_nb_continue($my_connection);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error downloading the file...";
  exit(1);
}
?>
```

Example #22 - Resuming a download at position 100 to a new file with ftp_nb_get()

```
<?php

// Disable Autoseek
ftp_set_option($my_connection, FTP_AUTOSEEK, false);

// Initiate
$ret = ftp_nb_get($my_connection, "newfile", "README", FTP_BINARY, 100);
while ($ret == FTP_MOREDATA) {

  /* ... */

  // Continue downloading...
  $ret = ftp_nb_continue($my_connection);
}
?>
```

In the example above, *newfile* is 100 bytes smaller than *README* on the FTP server because we started reading at offset 100. If we didn't disable **FTP_AUTOSEEK**, the first 100 bytes of *newfile* would be *'\0'*.

See Also

- ftp_nb_fget()
- ftp_nb_continue()
- ftp_fget()
- ftp_get()

ftp_nb_put

ftp_nb_put -- Stores a file on the FTP server (non-blocking)

Description

int **ftp_nb_put** (resource `$ftp_stream`, string `$remote_file`, string `$local_file`, int `$mode` [, int `$startpos`])

ftp_nb_put() stores a local file on the FTP server.

The difference between this function and the ftp_put() is that this function uploads the file asynchronously, so your program can perform other operations while the file is being uploaded.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The remote file path.

`local_file`
The local file path.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`startpos`

Return Values

Returns **FTP_FAILED** or **FTP_FINISHED** or **FTP_MOREDATA**.

Examples

Example #23 - ftp_nb_put() example

```
<?php

// Initiate the Upload
$ret = ftp_nb_put($my_connection, "test.remote", "test.local", FTP_BINARY);
while ($ret == FTP_MOREDATA) {
```

```
  // Do whatever you want
  echo ".";

  // Continue uploading...
  $ret = ftp_nb_continue($my_connection);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error uploading the file...";
  exit(1);
}
?>
```

Example #24 - Resuming an upload with ftp_nb_put()

```
<?php

// Initiate
$ret = ftp_nb_put($my_connection, "test.remote", "test.local",
                      FTP_BINARY, ftp_size("test.remote"));
// OR: $ret = ftp_nb_put($my_connection, "test.remote", "test.local",
//                            FTP_BINARY, FTP_AUTORESUME);

while ($ret == FTP_MOREDATA) {

  // Do whatever you want
  echo ".";

  // Continue uploading...
  $ret = ftp_nb_continue($my_connection);
}
if ($ret != FTP_FINISHED) {
  echo "There was an error uploading the file...";
  exit(1);
}
?>
```

See Also

- ftp_nb_fput()
- ftp_nb_continue()
- ftp_put()
- ftp_fput()

ftp_nlist

ftp_nlist -- Returns a list of files in the given directory

Description

array **ftp_nlist** (resource `$ftp_stream`, string `$directory`)

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`directory`
The directory to be listed. This parameter can also include arguments, eg. ftp_nlist($conn_id, "-la /your/dir"); Note that this parameter isn't escaped so there may be some issues with filenames containing spaces and other characters.

Return Values

Returns an array of filenames from the specified directory on success or **FALSE** on error.

Examples

Example #25 - ftp_nlist() example

```
<?php

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// get contents of the current directory
$contents = ftp_nlist($conn_id, ".");

// output $contents
var_dump($contents);

?>
```

The above example will output something similar to:

```
array(3) {
 [0]=>
 string(11) "public_html"
```

```
[1]=>
string(10) "public_ftp"
[2]=>
string(3) "www"
```

See Also

- ftp_rawlist()

ftp_pasv

ftp_pasv -- Turns passive mode on or off

Description

bool **ftp_pasv** (resource `$ftp_stream`, bool `$pasv`)

ftp_pasv() turns on or off passive mode. In passive mode, data connections are initiated by the client, rather than by the server. It may be needed if the client is behind firewall.

Please note that ftp_pasv() can only be called after a successfull login or otherwise it will fail.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`pasv`
If **TRUE**, the passive mode is turned on, else it's turned off.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #26 - ftp_pasv() example

```
<?php
$file = 'somefile.txt';
$remote_file = 'readme.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// turn passive mode on
ftp_pasv($conn_id, true);

// upload a file
if (ftp_put($conn_id, $remote_file, $file, FTP_ASCII)) {
echo "successfully uploaded $file\n";
} else {
```

```
echo "There was a problem while uploading $file\n";
}

// close the connection
ftp_close($conn_id);
?>
```

ftp_put

ftp_put -- Uploads a file to the FTP server

Description

bool **ftp_put** (resource `$ftp_stream`, string `$remote_file`, string `$local_file`, int `$mode` [, int `$startpos`])

ftp_put() stores a local file on the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The remote file path.

`local_file`
The local file path.

`mode`
The transfer mode. Must be either **FTP_ASCII** or **FTP_BINARY**.

`startpos`

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #27 - ftp_put() example

```
<?php
$file = 'somefile.txt';
$remote_file = 'readme.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);
```

```
// upload a file
if (ftp_put($conn_id, $remote_file, $file, FTP_ASCII)) {
echo "successfully uploaded $file\n";
} else {
echo "There was a problem while uploading $file\n";
}

// close the connection
ftp_close($conn_id);
?>
```

ChangeLog

Version	Description
4.3.0	*`startpos`* was added.

See Also

- ftp_pasv()
- ftp_fput()
- ftp_nb_fput()
- ftp_nb_put()

ftp_pwd

ftp_pwd -- Returns the current directory name

Description

string **ftp_pwd** (resource `$ftp_stream`)

Parameters

`ftp_stream`
The link identifier of the FTP connection.

Return Values

Returns the current directory name or **FALSE** on error.

Examples

Example #28 - ftp_pwd() **example**

```
<?php

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// change directory to public_html
ftp_chdir($conn_id, 'public_html');

// print current directory
echo ftp_pwd($conn_id); // /public_html

// close the connection
ftp_close($conn_id);
?>
```

See Also

- ftp_chdir()
- ftp_cdup()

ftp_quit

ftp_quit -- Alias of ftp_close()

Description

This function is an alias of: ftp_close().

ftp_raw

ftp_raw -- Sends an arbitrary command to an FTP server

Description

array **ftp_raw** (resource `$ftp_stream`, string `$command`)

Sends an arbitrary *`command`* to the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`command`
The command to execute.

Return Values

Returns the server's response as an array of strings. No parsing is performed on the response string, nor does ftp_raw() determine if the command succeeded.

Examples

Example #29 - Using ftp_raw() to login to an FTP server manually.

```
<?php
$fp = ftp_connect("ftp.example.com");

/* This is the same as:
  ftp_login($fp, "joeblow", "secret"); */
ftp_raw($fp, "USER joeblow");
ftp_raw($fp, "PASS secret");
?>
```

See Also

- ftp_exec()

ftp_rawlist

ftp_rawlist -- Returns a detailed list of files in the given directory

Description

array **ftp_rawlist** (resource `$ftp_stream`, string `$directory` [, bool `$recursive`])

ftp_rawlist() executes the FTP *LIST* command, and returns the result as an array.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`directory`
The directory path.

`recursive`
If set to **TRUE**, the issued command will be *LIST -R*.

Return Values

Returns an array where each element corresponds to one line of text.

The output is not parsed in any way. The system type identifier returned by ftp_systype() can be used to determine how the results should be interpreted.

Examples

Example #30 - ftp_rawlist() example

```
<?php

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// get the file list for /
$buff = ftp_rawlist($conn_id, '/');

// close the connection
ftp_close($conn_id);

// output the buffer
```

```
var_dump($buff);
?>
```

The above example will output something similar to:

```
array(3) {
 [0]=>
 string(65) "drwxr-x---   3 vincent  vincent      4096 Jul 12 12:16
public_ftp"
 [1]=>
 string(66) "drwxr-x---  15 vincent  vincent      4096 Nov  3 21:31
public_html"
 [2]=>
 string(73) "lrwxrwxrwx   1 vincent  vincent        11 Jul 12 12:16 www ->
public_html"
}
```

ChangeLog

Version	Description
4.3.0	*recursive* was added.

See Also

- ftp_nlist()

ftp_rename

ftp_rename -- Renames a file or a directory on the FTP server

Description

bool **ftp_rename** (resource `$ftp_stream`, string `$oldname`, string `$newname`)

ftp_rename() renames a file or a directory on the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`oldname`
The old file/directory name.

`newname`
The new name.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #31 - ftp_rename() example

```
<?php
$old_file = 'somefile.txt.bak';
$new_file = 'somefile.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to rename $old_file to $new_file
if (ftp_rename($conn_id, $old_file, $new_file)) {
echo "successfully renamed $old_file to $new_file\n";
} else {
echo "There was a problem while renaming $old_file to $new_file\n";
}

// close the connection
```

```
ftp_close($conn_id);
?>
```

ftp_rmdir

ftp_rmdir -- Removes a directory

Description

bool **ftp_rmdir** (resource `$ftp_stream`, string `$directory`)

Removes the specified *`directory`* on the FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`directory`
The directory to delete. This must be either an absolute or relative path to an empty directory.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #32 - ftp_rmdir() example

```
<?php

$dir = 'www/';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// try to delete the directory $dir
if (ftp_rmdir($conn_id, $dir)) {
   echo "Successfully deleted $dir\n";
} else {
   echo "There was a problem while deleting $dir\n";
}

ftp_close($conn_id);

?>
```

See Also

- ftp_mkdir()

ftp_set_option

ftp_set_option -- Set miscellaneous runtime FTP options

Description

bool **ftp_set_option** (resource `$ftp_stream`, int `$option`, mixed `$value`)

This function controls various runtime options for the specified FTP stream.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`option`
Currently, the following options are supported:

Supported runtime FTP options

FTP_TIMEOUT_SEC	Changes the timeout in seconds used for all network related functions. *`value`* must be an integer that is greater than 0. The default timeout is 90 seconds.
FTP_AUTOSEEK	When enabled, GET or PUT requests with a *`resumepos`* or *`startpos`* parameter will first seek to the requested position within the file. This is enabled by default.

`value`
This parameter depends on which *`option`* is chosen to be altered.

Return Values

Returns **TRUE** if the option could be set; **FALSE** if not. A warning message will be thrown if the *`option`* is not supported or the passed *`value`* doesn't match the expected value for the given *`option`*.

Examples

Example #33 - ftp_set_option() example

```
<?php
// Set the network timeout to 10 seconds
ftp_set_option($conn_id, FTP_TIMEOUT_SEC, 10);
?>
```

See Also

- ftp_get_option()

ftp_site

ftp_site -- Sends a SITE command to the server

Description

bool **ftp_site** (resource `$ftp_stream`, string `$command`)

ftp_site() sends the given *SITE* command to the FTP server.

SITE commands are not standardized, and vary from server to server. They are useful for handling such things as file permissions and group membership.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`command`
The SITE command. Note that this parameter isn't escaped so there may be some issues with filenames containing spaces and other characters.

Return Values

Returns **TRUE** on success or **FALSE** on failure.

Examples

Example #34 - Sending a SITE command to an ftp server

```
<?php
/* Connect to FTP server */
$conn = ftp_connect('ftp.example.com');
if (!$conn) die('Unable to connect to ftp.example.com');

/* Login as "user" with password "pass" */
if (!ftp_login($conn, 'user', 'pass')) die('Error logging into
ftp.example.com');

/* Issue: "SITE CHMOD 0600 /home/user/privatefile" command to ftp server */
if (ftp_site($conn, 'CHMOD 0600 /home/user/privatefile')) {
  echo "Command executed successfully.\n";
} else {
  die('Command failed.');
}
?>
```

See Also

- ftp_raw()

ftp_size

ftp_size -- Returns the size of the given file

Description

int **ftp_size** (resource `$ftp_stream`, string `$remote_file`)

ftp_size() returns the size of the given file in bytes.

Note
Not all servers support this feature.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

`remote_file`
The remote file.

Return Values

Returns the file size on success, or -1 on error.

Examples

Example #35 - ftp_size() example

```
<?php

$file = 'somefile.txt';

// set up basic connection
$conn_id = ftp_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

// get the size of $file
$res = ftp_size($conn_id, $file);

if ($res != -1) {
   echo "size of $file is $res bytes";
```

```
} else {
   echo "couldn't get the size";
}

// close the connection
ftp_close($conn_id);

?>
```

See Also

- ftp_rawlist()

ftp_ssl_connect

ftp_ssl_connect -- Opens an Secure SSL-FTP connection

Description

resource **ftp_ssl_connect** (string `$host` [, int `$port` [, int `$timeout`]])

ftp_ssl_connect() opens a SSL-FTP connection to the specified *`host`*.

Note
Why this function may not exist ftp_ssl_connect() is only available if OpenSSL support is enabled into your version of PHP. If it's undefined and you've compiled FTP support then this is why. For Windows you must compile your own PHP binaries to support this function.

Parameters

`host`
: The FTP server address. This parameter shouldn't have any trailing slashes and shouldn't be prefixed with *ftp://*.

`port`
: This parameter specifies an alternate port to connect to. If it is omitted or set to zero, then the default FTP port, 21, will be used.

`timeout`
: This parameter specifies the timeout for all subsequent network operations. If omitted, the default value is 90 seconds. The timeout can be changed and queried at any time with ftp_set_option() and ftp_get_option().

Return Values

Returns a SSL-FTP stream on success or **FALSE** on error.

ChangeLog

Version	Description
5.2.2	The function was changed to return **FALSE**

	when it can't use an SSL connection, instead of fallbacking to a non-SSL one as previously.

Examples

Example #36 - ftp_ssl_connect() example

```
<?php

// set up basic ssl connection
$conn_id = ftp_ssl_connect($ftp_server);

// login with username and password
$login_result = ftp_login($conn_id, $ftp_user_name, $ftp_user_pass);

echo ftp_pwd($conn_id); // /

// close the ssl connection
ftp_close($conn_id);
?>
```

See Also

- ftp_connect()

ftp_systype

ftp_systype -- Returns the system type identifier of the remote FTP server

Description

string **ftp_systype** (resource `$ftp_stream`)

Returns the system type identifier of the remote FTP server.

Parameters

`ftp_stream`
The link identifier of the FTP connection.

Return Values

Returns the remote system type, or **FALSE** on error.

Examples

Example #37 - ftp_systype() **example**

```
<?php

// ftp connection
$ftp = ftp_connect('ftp.example.com');
ftp_login($ftp, 'user', 'password');

// get the system type
if ($type = ftp_systype($ftp)) {
   echo "Example.com is powered by $type\n";
} else {
   echo "Couldn't get the systype";
}

?>
```

The above example will output something similar to:

```
Example.com is powered by UNIX
```