

WFTP API User Manual

Licence

WFTP was written by and is Copyrighted 1994 by Philippe Jounin. The author disclaims all liability for its use or for problems, data corruption, data loss, or other loss that may result from its use.

Permission is given without restriction to use and distribute the program provided it is distributed without charge, that it is not modified in any way, and that this file accompanies the DLL file.

My only wish is to receive a copy of any program which use the WFTP DLL.

Please send them to ark@ifh.sncf.fr.

Thanks To Santanu Lahiri for giving the source of WinFTP, a FTP client for windows. I have learn a lot (about FTP and Windows) by reading the source code.

Overview

WFTP.DLL provides an implementation of the FTP protocol (specified in the RFC 959).

It is a Windows Dynamic Library (DLL), which can be used by any language (and any compiler). It requires a Windows Sockets DLL (Winsock.DLL).

WFTP provides two groups of functions :

Asynchronous functions

Synchronous functions

The asynchronous functions return before the job is finished. The application will receive a message posted by the DLL when the job is over. The message contains two parameters wParam and lParam (please refer to a Windows programmer's reference) which are used to pass information such as return codes.

The functions return an integer which is TRUE (1) if the request is accepted, FALSE if it is rejected (in this case the application will receive no message).

The synchronous functions return when the job is finished. They return an integer which is directly the return code of the function.

The WFTP calls does not need any handle to identify the FTP session. They use the Windows function **GetCurrentTask** to get a task identifier. This mechanisms avoids the use of a parameter but it prohibits to have more than one FTP session for a given task (note that if the same application is started twice, WFTP will just see two different tasks, so each application can have its own FTP session).

Programming with the WFTP API

To use the WFTP functions 4 files are provided :

- This reference
- The DLL WFTP.DLL
- An include file WFTP.H
- A library file WFTP.LIB

The first function an application should call is **FtpInit**. It allocates buffers and get some information about the task which has called it.

The task is ready to make a connection with a FTP server. It must either

- Call **FtpOpenConnection**, **FtpSendUserName** and **FtpSendPasswd**
- or just call **AsyncFtpLogin** (which combines the 3 functions).

If it succeeds the user is logged on and can use any of the other WFTP functions. For Instance, the application can call **FtpDir** to read the contents of the remote directory.

To end the connection, the application must call **FtpCloseConnection**. If the function does **not** succeed (i.e. the network has been shutdown), it must call **FtpLocalClose**.

To release the allocated buffers, the application must call **FtpRelease** before it exits.

The WFTP functions

FtpInIt

FtpInIt must be call before any other function. It allocates buffers, read information about the task which has called it and create a invisible window for its internal use.

The function needs the handler of an application window.

Syntax : FtpInIt (HWND hParentWnd)

parameters : hParentWnd is the handle of an existing application window.

return codes :

FTPERR_OK	Initialisation has been done
FTPERR_INSMEMORY	not enough memory
FTPERR_CANTCREATEWINDOW	FtpInIt can't create its window
FTPERR_SESSIONUSED	The task has already a WFTP session

FtpSetVerboseMode

If a programmer wants to have a look on each frame sent by the server, he must use this function. He will get a message (by the **SendMessage** function) each time a frame has been received. The parameter wParam is TRUE, lParam points to the frame. It is nul terminated but can contains more than one line (a line is ended with <CR><LF>).

Note that the frame will be overwritten by the next reply from the server.

Syntax : FtpSetVerboseMode (BOOL bVerboseMode, WND hWnd, WMSG wParam)

parameters :	bVerboseMode	TRUE if the application wants to control incoming messages, FALSE to end a previous FtpSetVerboseMode
	hWnd	the handler of the windows to which the message is to be passed
	wParam	the application-defined message to be passed to the application each time a frame has been received.

return codes

FTPERR_OK	Mode has been changed
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit

FtpOpenConnection

This function establishes the connection with the FTP server. The connection port is the port 21. For this version of WFTP it can not be changed.

Once the connection is done, it waits for the reply of the server. The timeout is set to 30 secondes and can not be changed.

The reply must begin with "220" (RFC 959), if not a special error is given.

Syntax : FtpOpenConnection (LPSTR szHost)

Parameter : szHost : The name of the remote host to connect to

Return codes :

FTPERR_OK	Successful connection
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTCREATESOCKET	The socket has not been created
FTPERR_CONNECTREJECTED	Connection has been rejected (server is not a FTP server, ...)
FTPERR_CANTCONNECT	The connection has failed
FTPERR_TIMEOUT	The connection has timed-out
FTPERR_NOREPLY	The connection is succesful, but WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	The connection is succesful and WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connexion.

FtpSendUserName

This function sends the user's name to the server. This authentication is necessary to begin a file transfer.

Syntax : FtpSendUserName (LPSTR szUserName)

Parameter : szUserName : Name of the user

Return Codes :

FTPERR_OK	User is logged on
FTPERR_ENTERPASSWORD	Successful function but server awaits a password.
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connexion.

FtpSendPasswd

This function sends the password to the server.

Syntax : FtpSendUserName (LPSTR szUserName)

Parameter : szUserName : Name of the user

Return Codes :

FTPERR_OK	User is logged on
FTPERR_ENTERACCOUNT	Successful function but server awaits a account name
FTPERR_LOGINREFUSED	The USER/PASSWD has been rejected
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection.

FtpLogin

This function combines the three preceding functions. It complete the login procedure. It returns FALSE if the szHost parameter is NULL or if the session has not been initialized, else it returns TRUE. Once the request has been completed, the application will receive a wMsg message in the hWnd window.

The message will be followed by :

wParam : TRUE

lParam : The return code of the function

Syntax :

FtpLogin(LPSTR szHost,LPSTR szUser,LPSTR szPass, HWND hWnd,WMSG wMsg)

Parameters : szHost : name of the remote host (the computer on which the server is running)

szUser : name of the user

szPass : Password (it can be NULL if the user has no password)

hWnd is the handler of the windows to which the message is to be posted

wMsg is the application-defined message to be posted to the application

Return Codes :

Return codes are in the Low Word of the lParam parameter:

FTPERR_OK	User is logged on
FTPERR_ENTERACCOUNT	Successful function but server awaits an account name
FTPERR_LOGINREFUSED	The USER/PASSWD has been rejected
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection.
FTPERR_CANTCREATESOCKET	The socket has not been created
FTPERR_CONNECTREJECTED	Connect has been rejected (server is not a FTP server, ...)
FTPERR_CANTCONNECT	The connect has
FTPERR_TIMEOUT	The connect has timed-out

FtpCloseConnection

This function try to close gracefully the connection.It will not succeed if a file transfer is in progress or if the server has timed-out. You must then use FtpLocalClose.

Syntax : FtpCloseConnection (void)

Return Codes

FTPERR_OK	FTP session has been closed
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection.

FtpLocalClose

This function closes the opened socket without warning the server. You must use this function only if FtpCloseConnection has failed.

Syntax : FtpLocalClose (void)

Return Codes :

return FALSE if the session has not been initialized by FtpInit
else return TRUE.

FtpCWD

This function changes the default directory on the remote server.

Syntax : FtpCWD (LPSTR szPath)

Parameter : szPath : name of the new directory

Return Codes :

FTPERR_OK	Directory has been changed
FTPERR_SERVERCANTEXECUTE	CWD has failed (directory does not exists..)
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection

FtpDir

This function reads the remote directory. It can be used in two ways :

- The function posts a message to the application each time a file name is received.
- The function fills a file with the file names and posts a message once the directory is terminated.

In the first case, the function posts a message with wParam=FALSE each time a data line has been received. lParam is a pointer on this line. The application must save the data because the next line sent by the server will overwrite it. The string is nul-terminated and contains only one line (the ending <CR><LF> has been removed). The last message received by the application will have wParam=TRUE and lParam is the return code.

In the second case the dir is written in the file szFile. Once it is finished, WFTP posts a message to the application.

Syntax :

FtpDir (LPSTR szFilter, LPSTR szFile, BOOL bLongDir, HWND hWnd, WMSG wParam);

Parameters :

szFilter	Remote path and filename mask. Note that the wildcard expansion is dependent of the remote host and is not necessarily the same as MS DOS format. An empty string or NULL will give the current remote directory.
szFile	The file where the data is to be written, if szFile is NULL, the first mode is used (a message is posted each time a complete line has been received)
bLongDir	Allow the application to choose between the long or the short form of listing. The short form give only the name of the files, the format of the long form depends on the server.
hWnd	the handler of the windows to which the message is to be passed
wParam	the application-defined message to be passed to the application

Return Codes :

FTPERR_OK	Dir has been done
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_CANNOTCHANGETYPE	The server rejects the command TYPE ASCII
FTPERR_CANTOPENFILE	Local file can not be open
FTPERR_CANTWRITE	WFTP can not write in local file (disk full)
FTPERR_CANTCREATESOCKET	No more free sockets (Two sockets are needed)
FTPERR_TRANSFERREFUSED	the server refused the dir command
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection

FtpRecvFile

This function copies a remote file to a local file. The application will receive a message when the transfer is completed with wParam=TRUE (transfer completed), lParam=return code.

In the notification mode, the application will receive a message each time some data has been received. The same message as above is used but wParam will be FALSE, lParam will be the current position in the file (it is also the number of bytes which have been received).

Syntax :

```
FtpRecvFile (LPSTR szRemote, LPSTR szLocal,  
            char cType, BOOL bNotify,  
            HWND hParentWnd, UINT wParam)
```

Parameters :

szRemote	Remote file specification
szLocal	The file where to write the data.
cType	TYPE_A for ASCII, TYPE_B for binary
hWnd	the handler of the windows to which the message is to be passed
wParam	the application-defined message to be passed to the application

Return Codes :

FTPERR_OK	Dir has been done
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_CANNOTCHANGETYPE	The server rejects the command TYPE ASCII
FTPERR_CANTOPENFILE	Local file can not be open
FTPERR_CANTWRITE	WFTP can not write in local file (disk full)
FTPERR_CANTCREATESOCKET	No more free sockets (Two sockets are needed)
FTPERR_TRANSFERREFUSED	the server refused the Retrieve command
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection

FtpSendFile

This function copies a local file to a remote file. The application will receive a message when the transfer is completed with wParam=TRUE (transfer completed), lParam=return code.

In the notification mode, the application will receive a message each time some data has been sent. The same message as above is used but wParam will be FALSE, lParam will be the current position in the file (it is also the number of bytes which have been sent).

Syntax :

FtpSendFile (LPSTR szRemote, LPSTR szLocal,
char cType, BOOL bNotify,
HWND hWnd, UINT wParam)

Parameters :

szLocal	The file to be sent
szRemote	Remote file specification
cType	TYPE_A for ASCII, TYPE_B for binary
hWnd	the handler of the windows to which to pass the message
wParam	the application-defined message to pass to the application

Return Codes :

FTPERR_OK	Dir has been done
FTPERR_SESSIONNOTINITIALIZED	session has not been initialized by FtpInit
FTPERR_CANTSEND	WFTP can not send the data (network is down)
FTPERR_CANNOTCHANGETYPE	The server rejects the command TYPE ASCII
FTPERR_CANTOPENFILE	Local file can not be open
FTPERR_CANTWRITE	WFTP can not write in local file (disk full)
FTPERR_CANTCREATESOCKET	No more free sockets (Two sockets are needed)
FTPERR_TRANSFERREFUSED	the server refused the STOR command
FTPERR_NOREPLY	WFTP has received no reply. WFTP does not close the connection socket (use FtpLocalClose).
FTPERR_UNEXPECTEDANSWER	WFTP has received a reply. But this reply is not a valid FTP answer. WFTP does not close the connection

FtpGetFileSize

This function tries to get the size of the file which is to be received. It must be used immediately after a `FtpRecvFile`, because it searches in the last reply if the server has sent the size of the file.

If the function is successful, it will return the length of the file. (Note that in ASCII mode, it can be slightly different from the number of bytes WFTP will receive). else it returns 0.

Syntax

`FtpGetFileSize()`

returns `DWORD`.

FtpQuote

The last command for this version.

It allows the user to send to the server any command he wants. WFTP will send it to the server and waits for its reply.

The return code is either a FTP code (ie 200) or a WFTP error code (ie FTPERR_CANTSEND). The reply (if any) is copied into a user's buffer.

Syntax FtpQuote (LPSTR szCmd, LPSTR szReplyBuf, UINT uBufSize);

Parameters : szCmd The command to be sent
 szReplyBuf The buffer to copy the answer
 uBufSize The size of the user's buffer