Contents

What is SDFTP

SDFTP is a Script Driven FTP client. You specify the script filename and SDFTP executes the script line by line until it reaches the end. During the execution of a script SDFTP displays a typical looking file transfer type dialog box. SDFTP can be used to automate any number of routine FTP client operations. Just of a few of the things you can do with SDFTP are listed below.

- Download all of the files matching a wildcard pattern in a remote FTP server directory to a directory on your computer
- Download only the files you don't already have in a remote FTP server directory to a directory on your computer
- Download individually specified files
- Upload one file, some files or all files in a directory on your computer to a remote FTP server.
- Make and remove directories on a remote FTP server.
- Delete one file or all files matching a wildcard pattern in a remote FTP server directory.
- Delete files matching a wild card pattern on a directory on your computer, possibly after uploading them.
- Delay the execution of a script until a specified time.
- Connect to multiple FTP sites in one script.
- You can even do all of the above in one script.

You may specify what script to run in one of three ways:

- Place the name of the script on the command line after the program name
- Set up a file association for your script files and the SDFTP executable
- Simply execute SDFTP and it will ask you what script to run.

Please take a look at a few of the example scripts. The commands in the script are numbered and syntax is critical. SDFTP has a couple of instances where you will have to clear a message box to continue. SDFTP was designed to run completely silently and on it's own. Most messages can be suppressed. The messages that can not be suppressed are the Shareware reminders which are only in the Shareware version, and hard errors that absolutely could not have been predicted. For example a lost network connection error can be suppressed but running out of disk space or SDFTP not being able to find the script you specified can not be suppressed.

Commands

The following is a list of commands that you can use from within a SDFTP script. Click on the command to view more information about it.

Connection and Login/Logout Commands

CONN Connect to FTP Site
USER User name
PASS Password
ACCT Specify Account Information
QUIT Close this Connection

Transfer Commands

TYPE Transfer Type
PORT Establish a Data Port
STOR Store a File, Upload
MPUT Multiple Put, Upload
STOU Store Unique, Upload
APPE Append, Upload
RETR Retrieve a File, Download
MGET Multiple Get, Download
MGTC Multiple Get Conditional, Download

Directory Commands

MDIR Make Directory
RDIR Remove Directory

Delete Commands

DELL Delete Local File(s)
DELR Delete Remote File(s)

Miscellaneous Commands

LIST Get a Directory Listing
WTIL Wait Until (Hour and Minute)
EXEC Execute Program
CHWD Change Working Directory
UDEF User Defined

New Commands:

WFOR Wait For (Delay)
DIAL Establish/Close Dial Up Networking Session (32bit only)
GOTO Loop Around in your Script

New Section:

What to do if your FTP server does not use a / or \ to separate directories and files

Developers

The following is directed to developers who are calling SDFTP from their program. It deals with logging and how to know what is going on from within your program.

When a command is first processed SDFTP creates an entry in your script file under the [Status] heading. This entry is CS1= (Command Status), this entry directly corresponds to the command number being processed and is immediately set to F (Fail). If SDFTP completes the command successfully it will go back and set it to S (Success). After a script has been successfully executed once you will notice that the [Status] section has a CSx= entry for each command with the letter S following it. If these entries are left in the configuration file (which is fine), when SDFTP is executed with this same script it will set each and every CSx= entry to F before it does anything else. Then, as the script is processed SDFTP will set them to S if the command succeeds. One additional feature has been added to the Status section to make things a little easier on you. When SDFTP starts up it will create an entry under Status called ExitCode and will set it to be blank. If this entry already exist it will simply set it to be blank. If any error occurs during the execution of the script and SDFTP has to exit it will set this field to Failure otherwise when SDFTP reaches the end of the script and exits normally it will set this entry to Success. With all of that in mind it is possible that your program can call SDFTP with a script and simply monitor the Status/ExitCode entry for Success or Failure and if it reads Failure you can process the CSx= to see how far it got. You may find it better to set the Status/ExitCode entry to blank from your program so that there is no delay between your program monitoring it and SDFTP initializing it to be blank. One last bit of information on logging. Do not use a .log extension on your configuration file. SDFTP takes the name of your configuration file, strips off the extension and appends .log to it. This file contains everything read off of the control socket. In other words, it contains the FTP servers half of the conversation with you. Take a look at it, you may find it useful, and it will probably look familiar to a novice FTP user. A .err file is also created which contains your Winsock version and any error messages if your are in Silent mode.

Settings

The following is a list of settings that you can specify under the [Settings] entry in your script. A link to the sample scripts is also provided. Please note that most of the settings are the result of requests from people using SDFTP. They are typically designed to overcome an inability to perform a specific tasks. It is always possible that undesirable side effects will occur by using some of these settings.

Silent=1

- This setting tells SDFTP to write errors to the .err file instead of displaying a dialog box with the error message. The default behavior is to display a message box with the error information. The .err file is simply the name of your script file with the extension removed and .err added. When you are not in silent mode the .err file contains only your Winsock version information.

TimeOut=120

- This setting specifies what the SDFTP internal time out value should be. If you do not set this value it will default to 60 seconds. For scripts you run scheduled in the middle of the night you may want to increase this up to about 300 (5 minutes). The value you specify is in seconds.

OnSuccessExec=notepad

- This setting tells SDFTP to execute a program if it reaches the end of the script and no error has occurred. The options for this setting are the same as those for EXEC. The example given above will execute Notepad. You can specify the full path and command line options if needed.

OnFailExec=calc

- This setting tells SDFTP to execute a program if a failure occurs during the execution of the script. SDFTP will do all of it's cleanup and just before closing it will launch the specified program. The options for this setting are the same as those for EXEC. The example given above will execute Calculator. You can specify the full path and command line options if needed.

DebugMessages=1

- This setting will cause SDFTP to run in script debug mode. SDFTP will display a dialog box asking you to confirm each command just before it is sent to the FTP server. This can be very useful in determining why a script is not working.

Debug=1

- This setting causes SDFTP to include the client side of the control socket connection in the .log file. The .log file is simply the name of your script with the extension removed and .log added. By default it contains only the server side of the control socket connection. This file is typically only for your logging and debugging purposes.

NOgethostname=1

- This setting tells SDFTP not to execute the function gethostname() when it starts up. This function returns the name of your local computer. Some vendors winsock.dll take forever to return the local computers name. Some of them take forever to return and when they do return they still don't know what it is. For this reason you can implement this setting and SDFTP will fill in the local address field in the main window the first time you execute a CONN command instead of calling gethostname().

Blocking=1

- This settings tells SDFTP to use blocking calls on remote host name lookups. This was the default method in versions of SDFTP prior to v1.2.2.

No3D=1

- By default SDFTP will look for the clt3d.dll or ctl3d32.dll in your windows system directory. If it is found SDFTP will load and use the 3d controls for its dialog box. If it is not found, SDFTP will simply load and execute without it. The No3D=1 setting tells SDFTP not to even look for the dll just do without it. If you have versions of ctl3d.dll or ctl3d32.dll scattered throughout your system you may need this setting.

BringToTop=1
- When this setting is present SDFTP will bring itself to the top of the screen during the execution of a CONN statement. This feature was added because some 4GL's like to bring themselves to the top of the screen after launching SDFTP.

MPUTCanDoNothing=1
- If you execute an MPUT and specify a wildcard pattern that results in no files being found SDFTP will consider this to be an error and will immediately exit. This setting tells SDFTP to simply move on to the next command in the event that an error occurs getting a list of files from the wildcard pattern. I strongly recommend testing your scripts without this setting before you use it. SDFTP really does not know why a failure occurred when it tried to get a list of files it just knows that it did.

DELLCanDoNothing=1
- If you execute an DELL and specify a wildcard pattern that results in no files being found SDFTP will consider this to be an error and will immediately exit. This setting tells SDFTP to simply move on to the next command in the event that an error occurs getting a list of files from the wildcard pattern. I strongly recommend testing your scripts without this setting before you use it. SDFTP really does not know why a failure occurred when it tried to get a list of files it just knows that it did.

NoStoreFileError=1
- If your script tells SDFTP to upload a file that is stored on your computer and that file does not exist then SDFTP will exit with an error. This setting tells SDFTP to continue on with the next command.

Settings Dealing with Logging
Settings Dealing with File Deletion

Sample Scripts
What to do if your FTP server does not use a / or \ to separate directories and files

CONN

Usage:
CONN <FTP site name>:<remote port>

This command specifies what FTP site to connect to. If you place a : at the end of the site name and specify a remote port it will override the default port of 21.

Example:
1=CONN ftp.testsite.com
or
1=CONN ftp.testsite.com:34

Prerequisites to this command:
None

USER

Usage:
USER <username>

This command specifies what user name to log in as.

Example:
2=USER anonymous
or
2=USER testuser

Prerequisites to this command:

CONN

PASS

Usage:
PASS <password>

This command specifies the password for the user name you just specified. If you specify PASS without a password the program will prompt for it at run time. It is also worth mentioning that if you are calling SDFTP from your own program you could prompt the user for the password and then edit the script from within your program before you execute SDFTP.

Example:
3=PASS test@test.com
or
3=PASS

Prerequisites to this command:

CONN, USER

TYPE

Usage:
TYPE <A or I>

This command tells the remote FTP site that all transfers should be done in binary or ASCII mode. This command is not required because all FTP implementations default to ASCII transfers. If you are downloading binary files however you should set the TYPE to I which stands for Image which means binary. Once you set the TYPE in a script all transfers that follow will be done in that mode. The exception is if your script has more than one CONN entry in it. Then you should set the TYPE if necessary for each connection. The first example below sets the TYPE to ASCII and the second to Binary. In case this is unfamiliar to you, most of your zipped or self extracting programs and/or images should be done in binary mode, so set the TYPE to I.

Example:
4=TYPE A
or
4=TYPE I

Prerequisites to this command:

CONN, USER,PASS

PORT

Usage:
PORT

This command tells SDFTP to establish a port command and prepare for a transfer. This command must precede each and every individual upload or download and must be specified exactly as presented below. STOR, STOU, APPE, RETR, MGET, MGTC, LIST, and a DELR with a wildcard pattern also require a PORT command before they can be executed. The reason for this is that all of these commands require a data socket which is closed and destroyed after being used once. PORT creates a new data socket and tells the FTP server what it is.

Example:
5=PORT

Prerequisites to this command:

CONN, USER,PASS

RETR

Usage:
RETR <full path to remote file> <full path to local file to create>

This command specifies what file to download and what to call it on the local machine. This parameter starts downloading a file and storing it with the filename that you specify. The first part is the full path and filename to the file on the remote FTP server. The second part is the full path and filename to the file on the local machine that will be created as a result of this download.

Example:
6=RETR /pub/testdir/tesfile.zip c:\temp\testfile.zip
or
6=RETR /tesfile.zip c:\temp\newfile.zip

Prerequisites to this command:

CONN, USER,PASS,PORT

STOR

Usage:
STOR <full path to remote file to create> <full path to local file to upload>

This command specifies what file to upload and what to call it on the remote FTP server. This command starts uploading a file and storing it with the filename that you specify. The first part is the full path and filename to the new file on the remote FTP server. The second part is the full path and filename to the file on the local machine that will be uploaded to the FTP server.

Example:
8=STOR /testfile.zip c:\testfile.zip
or
8=STOR /incoming/newfile.zip c:\temp\testfile.zip

Prerequisites to this command:

CONN, USER,PASS,PORT

MGET

Usage:
MGET <full path to remote directory>\<wildcard pattern> <full path to local directory>

This command retrieves all files in a remote directory that match the wildcard pattern and stores them in the directory that you specify on your local machine. This command requires a PORT command before it. The full path including drive letter of where to put the files on your local computer is critical. You should specify at least a slash or backslash before the wildcard pattern when entering the remote directory location. SDFTP has to find one or the other so it knows what character to use to separate the directory that you specify with the filename the FTP server specifies. Please note that if a 550 error code is returned from the FTP server for one of the files to be downloaded, SDFTP will ignore it and move on to the next file. The 550 is typically returned for a file not found error. This will happen if the FTP server returns directory names with the list of files in the directory. SDFTP will send a retrieve command with the directory name specified and the FTP server will reply with something like "550 no such file". It is always possible that undesirable side effects will result with this behavior. I just have not seen any yet.

Example:
9=MGET /*.zip C:\temp\temp2
or
9=MGET /test/pub/* C:\temp\temp2\pub

Prerequisites to this command:

CONN, USER,PASS,PORT

What to do if your FTP server does not use a / or \ to separate directories and files

MGTC

Usage:
MGTC <full path to remote directory>\<wildcard pattern> <full path to local directory>

This command is exactly the same as MGET with one exception so be sure you understand MGET fully before using this command. The difference is simple, when executing this command SDFTP asks the remote FTP server for a list of files in the directory that you specified (same as MGET) however, the files in this list that already exist in your local destination directory will not be included in the download. Hence the name, MGET Conditional (MGTC). This can be used to keep a directory on your computer up to date with a directory on a remote FTP server. Filenames are the only thing that SDFTP compares, not dates, not sizes or anything else, just filenames. This has a beneficial side effect. Say you are FTP'ing a huge directory of large files. After a while you determine that you don't want half of them because they are taking up a lot of disk space but you still want to stay current with the FTP server so you think you can't delete them. Copy over the huge file with a small text file or something keeping the filename the same. This will cause SDFTP not to download it during a MGTC and you can have your disk space back. Remember this command requires a PORT command before it. The 16bit version of SDFTP will not correctly implement this command if the remote directory contains long filenames (not 8.3).

Example:
9=MGTC /*.exe C:\temp\temp2
or
9=MGTC /test/pub/* C:\temp\temp2\pub

Prerequisites to this command:
CONN, USER,PASS,PORT

What to do if your FTP server does not use a / or \ to separate directories and files

MPUT

Usage
MPUT <full path to remote directory> <full path to local directory>\<wildcard pattern>

This command uploads all of the files in your local source directory that match the specified wild card pattern to the directory specified on the remote FTP server. The trailing slash or backslash is required as well as the full path including drive letter for the local directory. The trailing slash or backslash for the remote destination directory tell SDFTP what character to use to separate the directory name from the filename.

Example:
9=MPUT / C:\temp\temp2*.*
or
9=MPUT /test/pub/ C:\temp*.zip

Prerequisites to this command:
CONN, USER,PASS,PORT

What to do if your FTP server does not use a / or \ to separate directories and files

MDIR

Usage:
MDIR <full path to remote directory to create>

This command creates a directory on the remote FTP server. The syntax is simply the full path to the directory to create.

Example:
9=MDIR /test/newdir
or
9=MDIR /test/newdir/newdir2

Prerequisites to this command:

CONN, USER,PASS

RDIR

Usage:
RDIR <full path to remote directory to remove>

This command removes a directory on the remote FTP server. The syntax is simply the full path to the directory to remove.

Example:
9=RDIR /test/newdir
or
9=RDIR /test/newdir/newdir2

Prerequisites to this command:

CONN, USER,PASS

DELL

Usage:
DELL <full path to local directory>\<wildcard pattern>

This command deletes all files in the directory that you specify on your computer if they match the wild card pattern. See the section on settings to remove the safety messages.

Example:
9=DELL C:\TEMP*.*
or
9=DELL C:\TEMP*.tmp

Prerequisites to this command:

None

DELR

Usage:
DELR <full path to remote directory>\<wildcard pattern or filename>

This command deletes all files matching a wildcard pattern or one file in the directory that you specify on the remote FTP server. If you specify a full path with a filename, that file will be deleted. If you specify a wildcard pattern then all files in that directory that match the wildcard pattern on the remote FTP server will be deleted. There is a special note about using the wildcard pattern. If you have a wildcard pattern at the end of the statement, SDFTP will ask for a list of all files in this directory. As it receives this list it builds a list of files to delete. The problem is that most FTP servers include directory names in this list. SDFTP will attempt to delete these as if they were files. The FTP server will reply to these with a 550 error message that means the specified file to delete was not found. There is no guarantee that all FTP servers will behave in this manner. If you have tested and you know that this is okay, SDFTP can be told to ignore these errors and continue. See the section on Deletion Settings. By the way, if the remote directory has no sub directories then this may not be an issue for you. This command when used with a wildcard pattern requires a PORTcommand before it.

Example:
9=DELR /test/script.log
or
9=DELR /test/*

Prerequisites to this command:

CONN, USER,PASS,PORT(If Using *)

What to do if your FTP server does not use a / or \ to separate directories and files

QUIT

Usage:
QUIT

This command closes your connection to an FTP server. This should always follow the last transfer you execute for each FTP server you connect to. The command has no options and should be specified as you see it below. This command does not cause SDFTP to close. SDFTP will close when there are no more commands to execute.

Example:
10=QUIT

Prerequisites to this command:

CONN, USER,PASS

WTIL

Usage:
WTIL <hour>:<minute>

This command tells SDFTP to wait right here until the hour and minute that you specify has arrived. The value is specified on a 24 hour clock so 7:00 p.m. is 19:00 and so on. When the time specified arrives, SDFTP will continue to the next command. The first example is 9:15 a.m. and the second is 10:30 p.m.

Example:
9=WTIL 9:15
or
9=WTIL 22:30

Prerequisites to this command:
None

Deletion Settings

The following three settings deal with file deletions.

DELR550OverRide=1

- This setting tells SDFTP that if you execute a DELR with one filename specified to delete and that file does not exist, do not consider this an error. By default SDFTP will consider attempting to delete a file that does not exist to be an error. This setting only applies to DELR.

DeleteOverRide=1

- This setting tells SDFTP not to prompt you for deletions while the script is running. With this option set SDFTP will assume that you know what you are doing and simply do what it was told without user intervention.

550DeleteOverRide=1

- When you execute a DELR with a * at the end this means delete all files in the remote directory. SDFTP will ask the remote FTP server for a list of files in that directory. With this list SDFTP will form individual delete commands to send to the server for each file. Most FTP servers return directory names as well as filenames when a file list is asked for. SDFTP has no way of telling directory names from filenames. SDFTP will send a delete command to the server with the directory specified and the server will typically respond with a "550 no such file" error. At this point SDFTP will check if this setting has been set. If it has not been set SDFTP will display an error and exit. If it has been set SDFTP will ignore it and continue. This setting applies only to this specific scenario. If the FTP server does not return directory names in the file list then this should not be an issue for you. If the server does return directory names then do not assume its behavior and set this setting, verify with a test directory that this delete command will return a 550 error before using this feature.

EXEC

Usage:
EXEC <options> <program to execute> <parameters to pass to program>

This command executes a program. When this command is processed, SDFTP reads in whatever you specify and attempts to execute is as a Windows program. Keep in mind that if a failure occurs before this command SDFTP will never process it (See the section on Settings). You may specify one of three options with this command. If you want the program to be executed minimized put -min before the program -max for maximized -hide for hidden. The default action is the launch the program in whatever state it considers to be normal.

Example:
9=EXEC notepad.exe c:\autoexec.bat
or
9=EXEC -min calc.exe

Prerequisites to this command:
None

Sample Scripts

Below is a list of sample scripts. All of the example scripts require modification because they use bogus values for the commands. The syntax remains correct.

Download One File

The following is an example of downloading one file from a remote FTP server.

[Settings]

[SDFTP]
1=CONN ftp.testserver.com
2=USER ftpuser
3=PASS mypassword
4=PORT
5=TYPE I
6=RETR /test/tempfolder/temp.zip c:\temp\tempfile.zip
7=QUIT

Download Multiple Files

The following is an example of downloading all files from a directory on a remote FTP server.

```
[Settings]

[SDFTP]
1=CONN ftp.somewhere.com
2=USER joeshmoe
3=PASS pretzel
4=TYPE I
5=PORT
6=MGET /test/tempfolder/* c:\temp
7=QUIT
```

Upload One File

The following is an example of uploading one file to a remote FTP server.

[Settings]

[SDFTP]
1=CONN ftp.server.net
2=USER myname
3=PASS mypassword
4=PORT
5=TYPE I
6=STOR /test/tempfolder/testfile.zip c:\temp\testfile.zip
7=QUIT

Upload Multiple Files

The following is an example of uploading multiple files to a remote FTP server.

```
[Settings]

[SDFTP]
1=CONN ftp.superserver.edu
2=USER johndoe
3=PASS baseball
4=PORT
5=MPUT /test/tempfolder/ c:\temp\*.*
6=QUIT
```

Delete One Remote File

The following is an example of deleting one file from a remote FTP server.

[Settings]

[SDFTP]
1=CONN ftp.deleteme.com
2=USER thedeleter
3=PASS itsgone
4=DELR /test/tempfolder/testfile.zip
5=QUIT

Delete All Files in One Remote Directory

The following is an example of deleting all of the files in one remote directory on the FTP server.

[Settings]

[SDFTP]
1=CONN ftp.servername.com
2=USER anonymous
3=PASS testuser@testserver.com
4=PORT
5=DELR /test/tempfolder/*
6=QUIT

Delete Local File(s)

The following is an example of deleting files from a directory on your computer.

[Settings]

[SDFTP]
1=CONN ftp.idontknow.com
2=USER anonymous
3=PASS anonuser@anonymous.com
4=DELL c:\temp*.zip
5=QUIT

Make and Remove a Directory

The following is an example of making a directory and removing a directory on the remote FTP server.

[Settings]

[SDFTP]
1=CONN ftp.superftp.edu
2=USER ftper
3=PASS ftppass
4=MDIR /test/tempfolder/newdir
5=RDIR /test/tempfolder/olddir
6=QUIT

A Little Bit of Almost Everything

The following is an example of using most of the script commands and settings in one script.

```
[Settings]
Silent=1
TimeOut=90
DebugMessages=1
550DeleteOverRide=1
DELR550OverRide=1
DeleteOverRide=1
OnSuccessExec=calc
OnFailExec=winmine

[SDFTP]
1=WTIL 23:45
2=CONN ftp.testserver.com
3=USER anonymous
4=PASS test@testserver.com
5=PORT
6=MGET /incoming/* c:\incoming
7=PORT
8=MPUT /outgoing/ c:\outgoing\*.*
9=PORT
10=DELR /incoming/*
11=DELL c:\outgoing\*.*
12=MDIR c:\outgoing\temp
13=PORT
14=STOR \outgoing\temp\tmpfile.zip c:\temp\tmpfile.zip
15=PORT
16=TYPE I
17=RETR \outgoing\temp\tmpfile.zip c:\temp\tmpfile2.zip
18=DELR \outgoing\temp\tmpfile.zip
19=RDIR \outgoing\temp
20=EXEC -max notepad
21=QUIT
22=CONN ftp.testsite2.com
23=USER testuser
24=PASS
25=PORT
26=MGTC /newfiles/* c:\incoming\newfiles
27=QUIT
```

How to Obtain the Registered Version

You can obtain the registered version of SDFTP by registering the program with RegNet. They offer a toll free 1-800 number for credit card orders as well as secure online ordering. Upon receipt of your registration the full registered version of SDFTP will be promptly e-mailed to your e-mail address as a file attachment.

A link to the authors e-mail address can be found at the Web page specified below.

A link to RegNet as well as the latest SDFTP program information can be found at the SDFTP home page:

http://www.netcom.com/~pgsoft

Please Note: The Shareware version of SDFTP has some limitations. It is the authors belief that the overall functionality of the program can be tested with the Shareware version. The licensed version of SDFTP is fully functional and is offered as a Personal Edition License and a Project Edition License. The following are the differences:
- The Personal Edition contains both SDFTP16 and SDFTP32 fully functional no nag versions. Displays the SDFTP version number along with Personal Edition on the title bar.
- The Project License contains both SDFTP16 and SDFTP32 fully functional no nag versions. Allows you to set the Title bar of the program to the name of your program. The Project License grants you redistribution rights for the inclusion of SDFTP into one of your programs. A single Project License will allow you to include SDFTP with your program and distribute it to 100 users. For every 100 users that will receive your program you will need one Project License. This license is intended for developers who need to call and monitor SDFTP, it is not a site license. Users who receive SDFTP as a result of receiving your program are not authorized to use it as a free standing program. Please contact the author for site licensing.

LIST

Usage:
LIST <full path to remote directory> <full path to local file to create>

This command specifies what directory you want a listing of followed by a filename to contain the directory listing on your computer. SDFTP will ask the FTP server for a directory listing and will store the complete list in the local file you specify. If you want a directory listing of your current directory specify a * as the remote directory. This command requires a PORT command before it as well as the transfer type to be ASCII. If you have not changed the transfer type in your script with a TYPE I then you are already in ASCII mode. If you have executed a TYPE I you will need to execute a TYPE A just before this command.

Example:
6=LIST /pub/testdir/ c:\temp\00index.txt
or
6=LIST * c:\temp\01index.txt

Prerequisites to this command:

CONN, USER,PASS,PORT,TYPE (Sometimes)

STOU

Usage:
STOU <full path to local file to upload>

This command specifies what file to upload to FTP server. If the remote FTP server supports STOU (which it does not have to) this command starts uploading a file and storing it with a filename assigned by the FTP server. The only parameter you need to specify is the local file to upload. The file will be placed in you current working directory. The filename assigned by the FTP server can be found in the .log file. The .log file is the name of your script with the extension removed and .log added.

Example:
8=STOU c:\testfile.zip
or
8=STOU c:\temp\testfile.zip

Prerequisites to this command:

CONN, USER,PASS,PORT,CHWD

APPE

Usage:
APPE <full path to remote file to create> <full path to local file to upload>

This command specifies what file to upload and what to append it to on the remote FTP server. This command starts uploading a file and storing it with the filename that you specify. If the file already exists on the FTP server the file you are uploading will be appended to the existing file. The first part is the full path and filename to the new file on the remote FTP server. The second part is the full path and filename to the file on the local machine that will be uploaded to the FTP server. The FTP server must support APPE for this command to work.

Example:
8=APPE /testfile.zip c:\testfile.zip
or
8=APPE /incoming/newfile.zip c:\temp\testfile.zip

Prerequisites to this command:

CONN, USER,PASS,PORT

ACCT

Usage:
ACCT <account information>

This command specifies what account to use.

Example:
3=ACCT admin
or
3=ACCT donotknow

Prerequisites to this command:

CONN, USER,PASS

UDEF

Usage:
UDEF <user defined command> <completion code>

This command specifies a command to send to the FTP server and the successful reply code that the FTP server will return. This command only applies to commands that require the FTP server to perform some function to itself and send a reply code back to the user. The example below renames a file on the remote FTP server. SDFTP will find the end of the string that you specify and backup 3 spaces, it will use the value found from here to the end as the completion code. This way you can be sure that all of your command is being sent to the FTP server. In the example below, renaming a file is a two step process. First you tell the FTP server what file you want to rename. The FTP server will replay with something like "350 file found, send new file name". At this point SDFTP grabs the 350 and continues. Next you send a command that says rename the file I just specified to this. The FTP server will now rename the file and send something like "250 file renamed" back to SDFTP. SDFTP will grab the 250 and continue to the next command.

Example:
5=UDEF RNFR test.zip 350
6=UDEF RNTO newtest.zip 250

Prerequisites to this command:

CONN, USER,PASS

CHWD

Usage:
CHWD <remote directory to change to>

This tells the FTP server to change the current working directory to the path you specify. This command should only be used in two instances. The first is if you need to execute a STOU (Store Unique) and you want the file to be created in a directory other than the one you log in to. The second is if your remote FTP server does not use a forward slash or a backslash to separate directories and files and you are trying to execute a MGET, MGTC, MPUT, or a DELR (with a *). Since some FTP servers use a period you will have to change your directory so that the commands do not add a slash or backslash. Follow the link below for more information on how to do this.

Example:
6=CHWD pub
or
6=CHWD ..

Prerequisites to this command:

CONN, USER,PASS

What to do if your FTP server does not use a / or \ to separate directories and files

What to do if your FTP server does not use a / or \ to separate directories from files

The goal of this section is to demonstrate how to use SDFTP with the CHWD command. One reason using CHWD is discouraged is because for the most part it is simply not needed. If you need the files in your current directory then a /* will usually work for the MGET, MGTC, and DELR. The other reason is that when SDFTP attempts to perform one of these commands that take a *, it asks the FTP server for a file listing of all the files in that directory. Here is the problem; some FTP servers return the full path to each file, others just the filenames. SDFTP has to find just the filename so it can append it to the directory that you specify and create individual retrieve or delete commands. The CHWD command does not necessarily change any of this it just makes it more difficult than it has to be. If the FTP server you are using uses a . instead of a / or \ SDFTP will not be able to separate the file list correctly unless we CHWD to the directory. The following are examples of wildcard type commands using this technique. Each example is a modified version of scripts that have helped other users in the past. A simple store (STOR) and retrieve (RETR) are not demonstrated here because they do not depend on the / or \ to upload or download files individually. DELR is not demonstrated because I can not verify its use in this situation. While it should be the same syntax as the first example, I have not been able to verify this so I am not going to submit to you that it works. If you need to try out a DELR look at the setting called DebugMessages=1. These examples are also assuming that when SDFTP executes a CHWD and then asks for a file list of the current directory the FTP server will return filenames only. If you believe your FTP server is not doing this let me know and I will look into some sort of setting for this.

Here is an example of MGET, MGTC (Multiple Get, Multiple Get Conditional)
This example will get all of the files in 'PUBLIC.NFS.TEMP' and put them in c:\temp. If you want to do a MGTC instead of MGET just change it to MGTC.

```
[SDFTP]
1=CONN ftp.testserver.com
2=USER ftpman
3=PASS pretzel
4=PORT
5=CHWD 'PUBLIC.NFS.TEMP'
6=MGET * C:\temp
7=QUIT
```

Here is an example of MPUT (Multiple Put)
This example will put all of the files in c:\temp in to 'PUBLIC.NFS.TEMP'

```
1=CONN ftp.testserver.com
2=USER ftpman
3=PASS pretzel
4=PORT
5=CHWD 'PUBLIC.NFS.TEMP'
6=MPUT * C:\temp\*.*
7=QUIT
```

Frequently Asked Questions

Please check the SDFTP Web Site for the absolute latest FAQ
http://www.netcom.com/~pgsoft

Q. How do I get the registered version of SDFTP?

A. See How to Obtain the Registered Verison

Q. What do I do if the FTP server does not use a / or \ and I want to use MGET?

A. See What to do if your FTP server does not use a / or \ to separate directories and files

Q. Does SDFTP support Firewalls or Passive Mode transfers?

A. No, but if you would like to see this support added, let us know.

Q. Why do I keep getting 425 errors that say something about a data connection?

A. You probably need to insert a PORT command before the command that is failing.

Q. How can I figure out what the completion code should be for a user defined command?

A. You can use a DOS box FTP session or some other FTP client to execute the command and see what the FTP server returns as a completion code. It will be a 3 digit code that tells you the command was successful.

Q. How come some times I get a progress bar during a retrieve and other times I don't?

A. If the FTP server tells SDFTP what the file size is then SDFTP will use it. If it does not then SDFTP will not display a progress bar because it does not have enough information to to do the math required to display a progress bar.

Q. Is there a limit to the size of a script?

A. Yes, 64kb. Everything beyond the 64kb limit is ignored. You can have somewhere in the neighborhood of 1000 commands in one script. It just depends on how many characters each command is.

Q. Why is the script a standard Windows ini file? Are you being lazy?

A. SDFTP was originally designed for developers to use. By using an ini file a developer of any windows programming language can access the script using a function set provided by Windows. This means that in order to use SDFTP the developer does not have to spend as much time developing routines to create and monitor scripts because the file I/O routines are a high level API built in to Windows.

Q. What possessed you to write this program and why do you keep enhancing it?

A. I ask myself the same question every day. But seriously, I saw a need and went with it

Q. What's next?

A. The next version will be 1.5.0 and will be 32bit only. It will add an HTTP download command to grab files off of the Web. In addition it will do away with command numbering. Version 1.5.0 will introduce some new developers interfaces as well as remove some of the old ones. Now is a great time for suggestions.

Logging Settings

The following four settings deal with SDFTP logging.

A file with an extension of .log and a filename the same as your script is created during SDFTP initialization. The .log file contains the servers side of you FTP conversation. Lines in this file that begin with *** are generated by SDFTP all other data is simply read from the FTP server. The exception to this rule is if you have set Debug=1, then the commands that SDFTP sends to the FTP server will be included in this file in the place in which they were sent.

LogFile=c:\logfiles\upload01.log
- This setting gives SDFTP a full path and filename for a log file to use. SDFTP defaults to creating a log file in the same directory as the script and named the same thing as the script with the extension removed and .log added.

AppendLogFile=1
- This setting tells SDFTP to append to the existing log file if it exists. If it does not exist it will be created. SDFTP defaults to creating a new .log file for each script each time it is executed.

A file with the extension .err and a filename the same as your script is created during SDFTP initialization. The .err file contains your Winsock version information followed by any errors that occurred during the execution of a script if you have set Silent=1. If you have not set Silent=1 then errors will be displayed with message boxes and this file will only contain you Winsock version information.

ErrorLogFile=c:\logfiles\upload01.err
- This setting gives SDFTP a full path and filename for an error log file to use. SDFTP defaults to creating a error log file in the same directory as the script and named the same thing as the script with the extension removed and .err added.

AppendErrorLogFile=1
- This setting tells SDFTP to append to the existing error log file if it exists. If it does not exist it will be created. SDFTP defaults to creating a new .err file for each script each time it is executed.

StatFile=c:\statfile\stat01.txt
- This setting tells SDFTP to create a statistic file of file transfer information. Every time a file transfer finishes SDFTP will total up the transfer statistics and put them in this well formatted text file.

AppendStatFile=1
- This setting tells SDFTP not to recreate the statistic file if it already exists. SDFTP will then append to the existing file.

Related:
See the section on Program Settings for information on Silent=1 and Debug=1

WFOR

Usage:
WFOR <minutes>:<seconds>

Wait For (WFOR) tells SDFTP to stay on this command until the specified amount of time has elapsed. If you only need to specify an amount in seconds then start with a colon and then the number of seconds. The first example will wait for 10 minutes and the second example will wait for 30 seconds.

Example:
6=WFOR 10:00
or
6=WFOR :30

Prerequisites to this command:

None

GOTO

Usage:
GOTO <command to go to>:<number of times to execute this command>

This command tells SDFTP to go to another command. If you only want to execute this loop a certain number of times, then place a colon after the command number to go to and follow the colon with the number of times you want this goto to execute. You can not have a GOTO within a GOTO. The first example will GOTO command 1 from command 6 forever. The second example will GOTO command 1 from command 6 a total of 5 times and then the GOTO will be skipped over.

Example:
6=GOTO 1
or
6=GOTO 1:5

Prerequisites to this command:

none
Note: This command will be removed with version 1.5.0

DIAL

Usage:
DIAL <Dial Up Networking Connection Name>,<Username>,<Password>
or
DIAL <Dial Up Networking Connection Name> -hangup

This command will cause the 32bit version of SDFTP to establish or close a DUN session. If the -hangup parameter is found then SDFTP will assume you want to close this connection. If you are already connected to the connection when SDFTP runs into the DIAL command, any subsequent calls in this script to close the connection will be skipped. This way you do not have to write one script for the times that you need to DIAL and one script for the times you do not need to DIAL. If the DIAL command detects that you are already connected then SDFTP will assume that you want to stay connected after the script is finished. The username and password are referring to your DUN username and password for the connection you specify. This command has only been tested under Win95 and NT 4.0. All values specified in this command are case sensitive. You may optionally specify any of the three parameters in quotation marks. You will need to do this if any of the three parameters have a comma in them.

Example:
6=DIAL my isp,myusername,mypassword
or
6=DIAL my isp,”123456,123456”,mypassword
or
6=DIAL my isp -hangup

Prerequisites to this command:

none