Using FTP from DataBasic

File Transfer Protocol (FTP) is a standard protocol for transmitting files between computers on a network. Like the Hypertext Transfer Protocol (HTTP), which transfers displayable Web pages and related files, and the Simple Mail Transfer Protocol (SMTP), which transfers e-mail, FTP is an application protocol that uses TCP/IP protocols. FTP is commonly used to transfer data files from their creator to the system where they will be used. It's also commonly used to download programs and other files to your computer from other servers.

There are utilities / applications that allow users to manually invoke file transfer, but automating the process requires a 'little' more effort. Using standard Reality features, file transfer can easily be automated.

Before we look at an example of how to invoke / use FTP we should cover some basics.

The Basics of FTP

Basic Order of Operations:

1. Change to your local directory where most (if not all) of the files you will be transferring are kept.
2. Open a connection to the remote host via the ftp command.
3. Once connected to the remote host, change to the directory (**cd** command) where the files are that you are going to get or to the location where you are going to put files.
4. Set the transfer mode (**ascii** or **binary**).
5. Transfer the files (**get**, **mget**, **put**, **mput**).
6. Repeat steps 1, 3, 4, 5 as necessary.
7. Exit ftp with the **bye** or **quit** command.

Commands:

- **ftp [*host*]** - open an ftp session with the specified *host* machine.

  Examples:
  C:\> ftp reality1
  C:\> ftp remote_system

- **open [*host*]** - Establish a connection to the specified *host* when you're already at an ftp prompt.

  Examples:
  ftp> open reality1
  ftp> open remote_system

- **user [*username*]** - Log into an ftp server when you're already connected in an ftp session.

  Examples:
  ftp> user sysadmin
  ftp> user anonymous

- **ls [*remote-directory*]** - Print a listing of the contents of *remote-directory* on the remote machine. The listing includes any system-dependent information that the server chooses to include.

  Examples:
  ftp> ls
  ftp> ls /usr/local/bin

- **dir [*remote-directory*] [*local-file*]** - Print a listing of the contents in the directory *remote-directory*, and optionally, placing the output in *local-file*.

Examples:
ftp> dir
ftp> dir /usr/local/bin

- **help [*command*]** - Print an informative message about the meaning of *command*. If no argument is given, ftp prints a list of the known commands.

  Examples:
  ftp> help
  ftp> help dir

- **?** - synonym for help.

  Examples:
  ftp> ?
  ftp> ? dir

- **pwd** - Print the name of the current working directory on the remote machine. Often this includes printing the full path.

  Example:
  ftp pwd>

- **cd [*remote-directory*]** - Change the working directory on the remote machine to *remote-directory*.

  Examples:
  ftp> cd /tmp
  ftp> cd ../..

- **lcd [*directory*]** - Change the working directory to *directory* on the local machine. If no directory is specified, the user's home directory is used.

  Examples:
  ftp> lcd c:\temp
  ftp> lcd ../..

- **ascii** - Set the file transfer type to ASCII . Only use this transfer method for text-files.

  Example:
  ftp> ascii

- **binary** - Set the file transfer type to support binary file transfer. Use this transfer method for anything other than a text file. For example, Word documents, .pdf files, .gifs, .jpgs, java class files, etc.

  Example:
  ftp> binary

- **put [*local-file*]** - Put (upload) *local-file* to the remote machine. No wildcards!

  Examples:
  ftp> put index.html
  ftp> put test.txt

- **get [*remote-file*]** - Retrieve (download) *remote-file* and store it on the local machine. No wildcards! Can only get one file at a time.

  Examples:
  ftp> get index.html
  ftp> get /tmp/readme.txt

- **mput [*local-files*]** - Expand wild cards in the list of *local-files* given as arguments and do a put for each file in the resulting list. The list of files should be separated by spaces.

  Examples:
  ftp> mput *
  ftp> mput *.html
  ftp> mput *.html test.txt README

- **mget [*multiple files and/or wildcards*]** - Expand wild cards in the list of remote files given as arguments and do a get for each file in the resulting list. The list of files should be separated by spaces.

  Examples:
  ftp> mget *
  ftp> mget *.gif
  ftp> mget *.doc image.gif salaries*

- **prompt** - Toggle interactive prompting. Interactive prompting occurs during multiple file transfers to allow the user to selectively retrieve or store files. If prompting is turned off, any mget or mput will transfer all files, and any mdelete will delete all files.

  Example:
  ftp> prompt

- **bell** - Arrange that a bell be sounded after each file transfer command is completed.

  Example:
  ftp> bell

- **delete [*remote-file*]** - Delete the *remote-file* on the remote machine.

  Examples:
  ftp> delete test.doc
  ftp> delete /tmp/temporary_file.txt

- **mkdir [*new-directory-name*]** - create a directory *new-directory-name* on the remote machine.

  Examples:
  ftp> mkdir temp
  ftp> mkdir /tmp/eric

- **rmdir [*directory-name*]** - Delete the directory entitled *directory-name* on the remote machine.

  Examples:
  ftp> rmdir temporary_directory
  ftp> rmdir /tmp/test_dir

- **rename [*old-file-name*] [*new-file-name*]** - Rename the file *old-file-name* on the remote machine, to the file *new-file-name*.

  Examples:
  ftp> rename index.htm homepage.html
  ftp> rename /tmp/readme.txt /tmp/README_NOW.txt

- **bye** or **quit** - Terminate the FTP session with the remote server and exit ftp. On Unix, an end of file should also terminate the session and exit.

  Example:
  ftp> bye

- **quote site chmod xxx [*file name*]** - Change the permission modes of the file *file-name* on the remote system to *xxx* mode. Note that the chmod command is not always implemented.

- **get [*file-name*] "|more"** - Instead of downloading and saving the file *file-name* on the local machine, you view its contents. Only recommended to use with text files.

Command-line options

ftp [-v] [-d] [-i] [-n] [-g] [-s:filename] [-a] [-w:windowsize] [computer]

- **-v** - Suppresses verbose display of remote server responses.
- **-n** - Suppresses auto-login upon initial connection.
- **-i** - Turns off interactive prompting during multiple file transfers.
- **-d** - Enables debugging, displaying all ftp commands passed between the client and server.
- **-g** - Disables filename globbing, which permits the use of wildcard chracters in local file and path names.
- **-s:filename** **(Windows only)** - Specifies a text file containing ftp commands; the commands will automatically run after ftp starts. No spaces are allowed in this parameter. Use this switch instead of redirection (>).
- **-a** - Use any local interface when binding data connection.
- **-w:windowsize** - Overrides the default transfer buffer size of 4096.
- **computer** - Specifies the computer name or IP address of the remote computer to connect to. The computer, if specified, must be the last parameter on the line.

To transfer data, using Reality, there are a few pre-requisites:

1. Access to the underlying OS.
2. Access to the remote system.
3. FTP is enabled for use.

These may seem to be simple enough, but can cause problems when implemented into a live environment.

Whether working in the Unix or Windows environment the amount of “Environment Dependency”, be it Unix or Windows, should be kept to a minimum. With this in mind, the following code example shows a simple method that has been proven to work.

Assume that a file (MYFILE.TXT) has been created using one of the variety of available methods, which can be employed using Reality (Sequential File Processing ; CSV-COPY ; DataBasic LISTSPREAD/SORTSPREAD ; COPY etc..), to create data that has to be transferred to another system.

The following example works within both Unix and Windows environments:

```
   IP_ADDRESS = "10.111.212.1"      ; * Remote system
   USER       = “valid_user_name”   ; * Remote system Logon Id
   PASSWORD   = “valid_password”    ; * Remote system password
   UNIX       = @FALSE
*******************************************
* This next piece of code assumes that the proprietary
* implementation of Reality no longer exists.
*******************************************
   IF SYSTEM(70) = 1 THEN
     LOCATION         = “/dumps”  ; * Unix world
     DELIM            = “/”
     POINT_TO_SCRIPT  = “< “
     UNIX             = @TRUE
   END ELSE
     LOCATION         = “C:\TEMP” ; * Windows world
     DELIM            = “\”
     POINT_TO_SCRIPT  = “-s:”
   END
   REMOTE.DIRECTORY = “/remotedirectory”
*******************************************
*  Setup a Reality pointer to the underlying OS directory
*******************************************
   PERFORM “DIR-VIEW FTPFILE “:LOCATION CAPTURING NULL
   FILENAME   = “MYFILE.TXT”
*******************************************
* Create a simple FTP script file.
*******************************************
   SCRIPTFILE        = “open “: IP_ADDRESS
   IF UNIX THEN
      SCRIPTFILE<-1> = “user “: USER :” “: PASSWORD
   END ELSE
      SCRIPTFILE<-1> = USER
      SCRIPTFILE<-1> = PASSWORD
   END
   SCRIPTFILE<-1>    = “lcd “: LOCATION
   SCRIPTFILE<-1>    = “cd “ : REMOTE.DIRECTORY
   SCRIPTFILE<-1>    = “binary”
   SCRIPTFILE<-1>    = “put “: FILENAME
   SCRIPTFILE<-1>    = “disconnect”
   SCRIPTFILE<-1>    = “quit”
*******************************************
   OPEN ‘FTPFILE’ TO FNAME ELSE STOP 201,”FTPFILE”
   WRITE SCRIPTFILE ON FNAME,”MYSCRIPT”
*******************************************
   PERFORM “sys ftp -inv “ : POINT_TO_SCRIPT : LOCATION : DELIM : ”MYSCRIPT”
   END
```