FTP Client Engine
Library for Xbase++

Programmer's Manual

(FCE4XB)

Version 4.0

November 23, 2023

This software is provided as-is.
There are no warranties, expressed or implied.

MarshallSoft Computing, Inc.
Huntsville AL 35815

Email: info@marshallsoft.com
Web: www.marshallsoft.com

TABLE OF CONTENTS

1 Introduction

The **FTP Client Engine for Xbase++** (FCE4XB library is a toolkit that allows software developers to quickly develop FTP client applications in Alaska Xbase++.

The **FTP Client Engine (FCE)** is a DLL library that uses the Windows API to provide direct and simple control of the FTP protocol. The FCE component library can be used for both anonymous and private FTP sessions.

A straightforward interface provides the capability to easily build Alaska Xbase++ software applications to connect to any FTP server, navigate its directory structure, list files, upload files, rename files, delete files, append files, and download files using the FTP protocol.

The **FTP Client Engine Programmers Manual** provides information need to compile and run programs in an Alaska Xbase++ programming environment.

The **FTP Client Engine for Xbase++** supports and has been tested with all versions of Alaska Xbase++ (Xbase++ v1.3 through Xbase++ v2.0). **FCE4XB** includes **s**everal example programs that demonstrate FTP processing from an Xbase++ application.

FCE4XB runs under all versions of 32-bit and 64-bit Windows through Windows 11.

The **FTP Client Engine SDK** DLL's (FCE32.DLL and FCE64.DLL) can also be used from any development environment (Visual Basic, C++, Delphi, Visual FoxPro, COBOL, Visual FoxPro, dBase, PowerBASIC, etc.) capable of calling the Windows API.

When comparing the **FTP Client Engine** library (FCE) against our competition, note that:

- FCE4XB is a standard Windows DLL (NOT an OCX or ActiveX control) and is much smaller than a comparable OCX or ActiveX control.
- FCE4XB does NOT depend on ActiveX or Microsoft Foundation Class (MFC) libraries or similar "support" libraries.
- FCE4XB is fully threadable.
- The FCE functions can be called from applications not capable of using controls.

MarshallSoft also has versions of the FTP Client Engine Library for C/C++ (FCE4C), Visual Basic (FCE4VB), Delphi (FCE4D), PowerBASIC (FCE4PB), Visual FoxPro (FCE4FP), and Visual dBASE (FCE4DB). All versions of FCE use the same DLLs (FCE32.DLL and FCE64.DLL). However, the examples provided for each version are written for the specified programming language.

All versions of the FTP Client Engine Library (FCE) can be downloaded from our web site at

http://www.marshallsoft.com/ftp-client-library.htm

1.1 Features

Some of the many features of the **FTP Client Engine** component library are as follows:

- Supports both 32-bit and 64-bit Windows.
- Connect to any (anonymous or private) remote FTP server.
- Get a list of files (names or long format) on the server.
- Navigate the server directories.
- Specify ASCII or BINARY transfer mode.
- Download files (with wildcard support).
- Upload files (with wildcard support).
- Delete files.
- Rename files.
- Append files.
- Create, delete and rename directories.
- Asynchronous and synchronous file transfer.
- Real-time upload/download data transfer rate.
- Real-time number bytes received/sent for async transfers
- Create and remove server directories.
- Support for PROXY servers.
- Supports active and passive mode (use with firewalls) file transfers.
- Supports multiple concurrent FTP sessions.
- Resume (restart) file uploads and downloads from any offset.
- Change files names while being uploaded or downloaded.
- Can parse long directory listings.
- Can specify the FTP or data port.
- Can set minimum and maximum response waits.
- Specify the allowable data port range.
- All operations can be aborted.
- Supports S/KEY password encryption.
- Use on Internet or your own intranet (LAN).
- Is native Windows code but can be called from managed code.
- Will run on machines without .NET installed
- Works with all versions of Xbase++ (v1.3 through v2.0).
- Does **not** depend on support libraries. Makes calls to core Windows API functions only.
- Can be used with any program (in any language) capable of calling Windows API functions such as Visual C++, .NET, Visual Basic, Delphi, dBASE, Visual FoxPro, COBOL, Access and Excel.
- Supports Windows through Windows 11.
- License covers all programming languages.
- Royalty free distribution with your compiled application.
- Can be purchased with or without source code [Ansi C].
- Updates are free for one year (Updates to source code are separate).
- Unlimited one-year email and phone support.
- Evaluation version is fully functional.

1.2 Documentation Set

The complete set of documentation consists of three manuals. This is the first manual (FCE4XB) in the set.

- FCE4XB Programmer's Manual (FCE_4XB.PDF)
- FCE User's Manual (FCE_USR.PDF)
- FCE Reference Manual (FCE_REF.PDF)

The FCE_4XB Programmer's Manual (FCE_4XB.PDF) is the language specific (Xbase++) manual dealing with compiler and programming issues such as installation and example programs. Read this manual first.

The FCE User's Manual (FCE_USR.PDF) discusses FTP in general as well as language independent programming issues such as application notes and includes purchasing and license information. Read this manual second.

The FCE Reference Manual (FCE_REF.PDF) contains details on each individual FCE function.

All manuals can also be viewed online at http://www.marshallsoft.com/fce4xb.htm.

1.3 Using the FCE Library

The **FTP Client Engine** component library **(FCE)** has been tested on multiple computers running 32-bit and 64-bit Windows through Windows 11.

The FCE4XB library has been tested with all versions of 32-bit Alaska Xbase++. The FCE32.DLL functions may be called by any Windows application program capable of calling the Windows API provided that the proper declaration file is used. FCE64.DLL is available to use with Win64 applications.

The SETUP installation program will copy the DLL to the Windows directory. Refer to Section 1.6 "Installation". After SETUP is run, the FCE4XB files are copied to the directory specified (default \FCE4XB). Three sub-directories are created, as follows:

```
DOCS - All documentation files
APPS - All example code
DLLS - All DLL's
```

1.4 Error Display

The error message text associated with FCE error codes can be displayed by calling **fceErrorText**. Each sample program contains examples of error processing.

Also see the file fceErrors.txt for a list of all FCE error codes

1.5 Example Program

The following example demonstrates the use of some of the **FTP Client Engine** library functions:

```
#INCLUDE "DLL.CH"
#INCLUDE "KEYCODE.CH"
#INCLUDE "FCE32.CH"

#define TEMP_SIZE 128
#define DATA_SIZE 5000

Procedure Main()
FTPserver = "ftp.drivehq.com" + Chr(0)
FTPuser = "anonymous" + Chr(0)
FTPpass = "you@gmail.com" + Chr(0)
FTPdirectory = "/MarshallSoft/PublicFolder" + Chr(0)
FTPfilename = "fceErrors.txt" + Chr(0)
* attach FCE
Code = XfceAttach(1, FCE_KEY_CODE)
if Code < 0
  ? "Cannot attach FCE ", Code
  return
endif
* connect to FTP server
? "Connecting to ", FTPserver
Code = XfceConnect(0, FTPserver, FTPuser, FTPpass)
if Code < 0
  DisplayError(Code)
  Code = XfceRelease()
  return
endif
? "Connected. Now downloading file..."
* change to proper directory
Code = XfceSetServerDir(0, @FTPdirectory)
* set to ASCII xfer mode
Code = XfceSetMode(0, ASC("A"))
* download the file
Code = XfceGetFile(0, @FTPfilename)
if Code <= 0
   DisplayError(Code)
else
   ? "File downloaded."
endif
* log off
Code = XfceClose(0)
Code = XfceRelease()
? "Logged off."
return

Procedure DisplayError(ErrCode)
TempBuffer = SPACE(TEMP_SIZE)
Code = XfceErrorText(0,ErrCode,@TempBuffer,TEMP_SIZE)
? Left(TempBuffer,Code)
return
```

In the example program above, **XfceConnect** is called to connect to the FTP server as user "anonymous" and password "msc@traveller.com". Note that functions defined for Xbase++ begin with '**X**'. All strings passed to FCE functions must be prefixed with the '**@**' character.

The server directory is changed to "pub/other", the transfer mode is set to ASCII, and the file "fce-new.txt" is downloaded. Lastly, the connection to the FTP server is closed and FCE is released.

Refer to the FCE Reference Manual (FCE_REF) for individual function details. Access online at http://www.marshallsoft.com/fce_ref.pdf

1.6 Installation

(1) Before installation of FCE4XB, an Xbase++ compiler (any version) should already be installed on the system and tested.

(2) Unzip FCE4XB40.ZIP (demo version) or FCExxxx.ZIP (registered version; xxxx is the Customer ID).

(3) Run the installation program SETUP.EXE which will install all FCE4XB files, including copying FCE32.DLL to the Windows directory, which is normally C:\WINDOWS.

(4.) You are ready to compile and run! For a quick start, load the project file FCEVER.PRG.

1.7 Uninstalling

Uninstalling FCE4XB is very easy.

First, run UINSTALL.BAT, which will delete FCE32.DLL from your Windows directory, normally C:\WINDOWS.

Second, delete the FCE4XB project directory created when installing FCE4XB.

1.8 Pricing

A developer's license for FCE4XB can be registered for $119 ($299 with ANSI C source code to the FCE DLL). Purchasing details can be found in Section 1.3, "How to Purchase", of the FCE User's Manual (FCE_USR).

Registration includes one year of free updates and technical support.

1.9 Updates

When a developer license is purchased for FCE4XB, the developer will be sent a registered DLL plus a license file (FCExxxx.LIC, where xxxx is your Customer ID). The license file can be used to update the registered DLL for a period of one year from purchase. Updates can be downloaded from

http://www.marshallsoft.com/update.htm

After one year, the developer license must be updated to be able to download updates and receive technical support. The license can be updated for:

- $33 if the update is ordered within one year of the original purchase (or previous update).
- $55 is the update is ordered between 1 and 3 years of the original purchase (or previous update).
- $77 if the update is ordered after three years of the original purchase (or previous update).

The update price includes technical support for an additional year. Note that the registered DLLs, (FCE32. DLL and FCE64.DLL) never expire. If source code was previously purchased, updates to the source code can be purchased for $100 along with the license update. Refer to the file UPDATES.TXT located in the /FCE4DB/DOCS directory for more information.

Note that the registered DLL never expires.

2 Library Overview

2.1 Keycode

FCE32.DLL has a keycode encoded within it. The keycode is a 9 or 10-digit decimal number (unless it is 0), and will be found in the file KEYCODE.CH. The keycode for the evaluation version is 0. The developer will receive a new key code after registering. SETUP copies the keycode to the /APPS sub-directory (folder). The KEYCODE is passed to **fceAttach**.

If the error message (value -74) is returned when calling **fceAttach**, it means that the keycode in the application does not match the keycode in the DLL. After registering, it is best to remove the evaluation version of the FCE DLL from the Windows search.

2.2 Dynamic Link Libraries

The **FTP Client Engine** library is implemented as a dynamic link library (DLL). A Win32 DLL is included. A DLL is characterized by the fact that it need not be loaded until required by an application program and that only one copy of the DLL is necessary regardless of the number of application programs that use it. Contrast this to the traditional static library that is bound to each and every application that uses it at link time.

An important advantage that DLL's have over other "popular" library formats such as VBX or OCX is that DLL's are callable by all Windows applications. Since DLL's are the building blocks of the Windows Operating System, they will not be replaced by a "newer technology".

2.3 Win32 STDCALL and DECLSPEC

FCE32 is written in ANSI C and is compiled using the STDCALL and DECLSPEC keywords. This means that FCE4XB uses the same calling conventions and file naming conventions as the Win32 API. In particular, function names are NOT decorated. There are no leading underscores nor trailing "@size" added to function names.

The FCE32.DLL functions may be called by any Windows application capable of calling the Windows API provided that the proper declaration file is used.

2.4 FTP Parameters

There are two types of FTP connections: private and anonymous. However, some FTP servers do not accept anonymous connections.

Three parameters are necessary in order to connect to an FTP server, as follows:

- Host name (or IP address) of the FTP server.
- User name.
- User password.

For private connections, the users account name and password must be specified.

For anonymous connections, the user name is "anonymous" and the password is the user's email address.

These FTP parameters are hard coded in most of the examples. However, these parameters could be read from the keyboard, from a file, from a dialog box at runtime, etc., as well as being hard coded.

Refer to the FCE User's Manual (FCE_USR) for more information regarding FTP protocol parameters.

2.5 FCE Declaration File

All FCE functions are declared in the FCE declaration file FCE32.CH. This file can be copied to the Xbase++ INCLUDE directory (where Xbase++ can find it) if so desired.

Note that each function is declared with the prefix character of 'X'.

2.6 Dynamic Strings

A string in the C language (in which the FCE and Windows are written) consists of a pointer to the first byte of a character buffer in which a zero byte ends the string characters.

When passing a string to a FCE function, it is best to always append the Chr(0) character to the end of the string. This will guarantee than the FCE functions will be able to detect the end of the string. The example in Section 1.5 above illustrates this.

When passing a string buffer to a DLL function into which text will be copied, it is strongly recommended that the local string be allocated immediately before use. For example, a string buffer is passed to the user defined dllGetMessage function, which copies a text message into it. Note that `SPACE$(80)` is called immediately before dllGetMessage.

```
* allocate buffer just before call to dllGetMessage
Buffer = SPACE(80)
Code = dllGetMessage(@Buffer, 80)
* message text is now in 'Buffer'
```

This technique is not necessary for passing a string to a DLL function, only when passing a buffer to a DLL into which data is to be placed by the DLL function.

2.7 Adding FCE to a Project

It is straightforward to add FCE to Xbase++ programs. First, add

```
#INCLUDE "KEYCODE.CH"
#INCLUDE "FCE32.CH"
```

after any other $INCLUDE statements in the Xbase++ program.

Then add

```
nCode = fceAttach(1, FCE_KEY_CODE)
If nCode < 0 Then
  ? "Cannot attach FCE"
  return
endif
```

as the first executed FCE function.

The keycode (contained in KEYCODE.CH) is 0 for the evaluation version and is a 9-digit number for the purchased version. Rather than include KEYCODE.CH as shown above, the keycode can be pasted directly into the call to fceAttach.

Lastly, link your program with FCE32.LIB. Refer to the example programs in the APPS subdirectory.

3 Compiler Issues

3.1 INCLUDE Files

All example programs include two files; `KEYCODE.CH` and `FCE32.CH`. The file `FCE32.CH` contains all the necessary constants and function declarations for FCE4XB, while the file `KEYCODE.CH` contains your key code, as discussed in Section 2.1.

The Xbase++ include file DLL.CH is also required. For example,

```
#INCLUDE "DLL.CH"
#INCLUDE "KEYCODE.CH"
#INCLUDE "FCE32.CH"
```

3.2 Compiling and Linking Programs

Before compiling any of the example programs, edit each program with your FTP user parameters, as shown in the example program in Section 2.4 above. Server names can be IP addresses (in decimal dot notation) or the host name.

For more information on host names and FTP user parameters, refer to the FCE User's Manual (FCE_USR). See Section 4.0 below, "Example Programs", for more details on each of the example programs.

To compile and link console mode programs such as LIST.PRG, use:

```
xpp list.prg
alink /subsystem:console list.obj fce32.lib
```

To compile and link windows GUI programs, such as WINFTP.PRG, use:

```
xpp winftp.prg
alink /subsystem:windows winftp.obj fce32.lib
```

3.3 Xbase++ Compiler

If you don't have the Alaska Software Xbase++ compiler, you can find it on the web at

```
http://www.alaska-software.com
```

4 Example Programs

All example programs will work with all versions of Alaska Xbase++. Each has been tested and shows how to correctly use FCE functions. It suggested that the developer compile and run the example programs before developing an application using FCE4XB
.
Each example program, with the exception of FCEVER.PRG and WINFTP.PRG, must be edited with your FTP protocol parameters (Section 2.4) before compiling.

Refer to Section 3.2 above for information on compiling and linking the example programs.

4.1 FCEVER

The first example program is the program FCEVER (FCE Version) that displays the FCE library version number and registration string. Its purpose is to display the current version of the FCE DLL as well as to verify that FCE32.DLL is being found and loaded by Windows. It does not require a TCP/IP (Internet) connection.

```
xpp fcever.prg
alink /subsystem:console fcever.obj fce32.lib
```

4.2 GET

The GET example program connects to our FTP server anonymously and downloads the file "fce-new.txt".

```
xpp get.prg
alink /subsystem:console get.obj fce32.lib
```

4.3 LIST

The LIST example program logs onto the specified FTP server and lists all files. Edit LIST.PRG with the FTP server name (or IP address), user name, and password before compiling.

```
xpp list.prg
alink /subsystem:console list.obj fce32.lib
```

4.4 FIELDS

The FIELDS example program logs onto the specified FTP server and lists all files by field. Edit FIELDS.PRG with the FTP server name (or IP address), user name, and password before compiling.

```
xpp fields.prg
alink /subsystem:console fields.obj fce32.lib
```

4.5 FTP

The FTP example program is a command line driven FTP client. Edit FTP.PRG with the FTP server name (or IP address), user name, and password before compiling

```
xpp ftp.prg
alink /subsystem:console ftp.obj fce32.lib
```

4.6 FTP2

The FTP2 example program is the same as the FTP.PRG example, except that it uses the FCE direct mode rather than FCE indirect mode. Refer to Section 4, "Theory of Operation", in the FCE User's Manual (FCE_USR).

Edit FTP.PRG with the FTP server name (or IP address), user name, and password before compiling

```
xpp ftp2.prg
alink /subsystem:console ftp2.obj fce32.lib
```

4.7 WINFTP

The WINFTP example program is a Windows GUI example rather than a Windows console mode program. The Forms Designer (XPPFD.EXE) was used to create the forms code, after which the FCE code was added.

WINFTP is a general purpose FTP client. The WINFTP application can be used to connect to an FTP server and upload, download, and delete files. All necessary FTP parameters are entered at runtime.

```
xpp winftp.prg
alink /subsystem:windows winftp.obj fce32.lib
```

4.8 PROXY

The PROXY example program connects to an FTP server through a proxy server using the "USR@SERVER" protocol. Edit PROXY.PRG with your FTP server name (or IP address), user name, and password before compiling.

For a discussion of proxy servers, refer to Section-3.7 “Proxy Server", of the FCE User’s Manual (FCE_USR).

```
xpp proxy.prg
alink /subsystem:console proxy.obj fce32.lib
```

4.9 MGET

The MGET example program downloads files according to a wildcard pattern (using '?' and '*' characters). Edit MGET.PRG with your FTP server name (or IP address), user name, and password before compiling.

```
xpp mget.prg
alink /subsystem:console mget.obj fce32.lib
```

4.10 MPUT

The MPUT example program uploads files according to a wildcard pattern (using '?' and '*' characters). Edit MPUT.PRG with the FTP server name (or IP address), user name, and password before compiling.

Note that most FTP servers do **not** allow anonymous users to upload files.

```
xpp mput.prg
alink /subsystem:console mput.obj fce32.lib
```

4.11 SPEED

The SPEED example program connects to the MarshallSoft FTP server at ftp://ftp.marshallsoft.com and downloads a test file from the MarshallSoft FTP server and displays the time required. Use this program to see how long it takes to download files from FTP servers.

```
xpp speed.prg
alink /subsystem:console speed.obj fce32.lib
```

4.12 FTIME

The FTIME example program requests the "File Modification Time" for a file. However, not all FTP servers support the FTP MDTM command used to request the file modification time.

```
xpp time.prg
alink /subsystem:console time.obj fce32.lib
```

5 Revision History

The FTP Client Engine DLLs (FCE16.DLL and FCE32.DLL) are written in ANSI C. All language versions of FCE (C/C++, Delphi, Xbase++, PowerBASIC, FoxPro, dBase, Xbase++, and COBOL) use the same identical DLLs.

Version 2.0: May 18, 2000

- The initial release of the Xbase++ version of FCE.

Version 2.1: February 13, 2001.

- Increased buffer size from 64 to 128 bytes for LocalDir and ServerDir.
- WriteBufferSize default increased to 512.
- Password not used if specified password has zero length.
- fceSetLocalDir() verifies that local directory is writable.
- Added FCE_SET_APPEND_MODE.
- Allow 128 character filenames.
- Added FCE_RENAME_DELIMITER to fceSetInteger().
- Corrected fceGetStatus(Chan, FCE_CONNECT_STATUS)
- Added FCE_SET_CLIENT_OFFSET and FCE_SET_SERVER_OFFSET.
- Rename file being up/downloaded by specifying "oldname:newname" for file.
- Added FCE_GET_LOCAL_IP to fceGetString.

Version 2.2: October 25, 2001.

- Default write buffer size increased from 512 to 1024 (WIN32 only).
- Added fceMatchFile function (used in multi-file uploads and downloads).
- Specify "User" as Chr(0) [in fceConnect] to skip USER and PASS processing.
- Performance improvements.
- Added MGET and PROXY example programs.

Version 2.3: December 20, 2002.

- Added FCE_SET_BLOCKING_MODE to control blocking (default ON) while connecting.
- Size of command buffer in fceCommand increased from 64 to 128 bytes.
- "WARNING: 226/250 not seen" written to log file rather than returning error.
- Added FCE_GET_ERROR_LINE.
- fceCommand sends CRLF with command in one network write.
- Added fceFileLength function.
- fceExtract handles line # 0.
- Added FCE_GET_QUEUE_ZERO (returns # times fceQueueLoad returns 0).
- Fixed problem with long server offsets (replaced "REST %d" with "REST %ld") in 16-bit version.
- Changed to 32 channels & 128 data ports.
- fceGetList returns error if receive buffer is too small.

Version 2.4: June 22, 2004.

- Added FCE_SET_CONNECT_WAIT_IN_SECS.
- Added FCE_SET_MAX_RESPONSE_WAIT_IN_SECS.
- Added FCE_SET_MAX_LINE_WAIT_IN_SECS.
- FCE_NOT_COMPLETED returned if code 226 (or 250) not returned by control socket (in 2 tries).
- Number data ports changed to 2048 per channel for 1 and 2 channels
- Number data ports changed to 512 data ports per channel for 3 to 8 channels
- Number data ports changed to 128 data ports per channel for 9 to 32 channels
- DataPort mask corrected to 0x7FFF
- Added fceGetLocalFList function to get list of files in local directory.
- Added fceGetLocalFSize function to get the size of a file in the local directory.
- Added new MPUT and MGET example programs (using wildcards).
- Added ShowProgess function to WINFTP.PRG

Version 2.5: July 26, 2005.

- LocalDir always stored with backslash as last character.
- fceWriteSocket makes up to 12 attempts to write.
- Add FCE_AUTO_LOG_CLOSE and FCE_CLOSE_LOG_FILE
- Adjusted wait time-outs.
- Added FCE_NO_GREETING error.
- Improved operation of fceGetInteger(Chan, FCE_GET_CONNECT_STATUS).
- Added GET example program.

Version 2.6: February 27, 2007

- Maximum PUT buffer size increased from 8K to 16K (16384)
- Recoded sleep wait in fceWriteSocket for improved upload performance.
- Added internal memory allocation debugging.
- Added GET_FULL_RESPONSE
- Close control socket whenever fceConnect fails (fixes socket leak problem).
- Added FCE_SET_FIRST_DATA_PORT and FCE_SET_LAST_DATA_PORT to fceSetInteger.
- Maximum data port extended to 65535.
- Added FCE_HIDE_PASSWORD to fceSetInteger.
- Added S/KEY authentication
- Added fceGetTicks()
- Added FCE_STATUS_BEFORE_WRITE to fceSetInteger.

Version 2.7: July 23, 2008

- Fixed problem with non-blocking mode when connecting.
- Added FCE_LOCAL_DIR_IS_CDROM to fceSetInteger, which allows the local directory to be a read-only device such as a CDROM.
- Added FCE_DISABLE_SKEY to fceSetInteger, which disables S/KEY processing.
- Added MDTM example program.

Version 3.0: October 22, 2009

- Added support for 64 bit Windows (FCE64.DLL)
- Added fceIsConnected function
- Added fceToInteger function

Version 3.1: July 13, 2011

- Log file is now time-stamped
- Added diagnostics to fceFileLength
- Function fceGetLocalFList no longer counts subdirectories
- Added fceGetFileSize function
- Added fceGetFileTime function

Version 3.2: June 7, 2012

- Fixed Bug: Open control socket not always closed.
- Fixed Bug: Open listen socket not always closed.
- Added function fcePutDirFiles (uploads all files in directory)
- Added function fceGetDirFiles (downloads all files in directory)

Version 3.3: April 30, 2014

- Fixed problem with connecting w/o blocking.
- Fixed problem with server name being corrupted in the FCE log file.
- Added function fceGetSubDirs()
- Added FCE_SET_DEBUG_LEVEL to fceSetInteger().

Version 3.4: November 19, 2015

- Added debug diagnostics to fceSocketStatus().
- Fixed problem with connecting w/o blocking.
- Fixed problem with server name being corrupted in the FCE log file.
- Replaced FCE_EOF with FCE_CANNOT_OPEN if FCE log file cannot be created.
- Automatically adjust sleep times for slow FTP servers.

Version 4.0: November 23, 2023

- Fixed problem: In some cases FCE functions returned 1 instead of -1
- Log file displays filename passed to fceGetFile and fcePutFile
- Data no longer written to log file
- Default write size buffer increased from 1024 to 4096
- Default is set to ASCII mode (as opposed to binary)
- Added "Not connected to server" error message.(FCE_NOT_CONNECTED)
- I/O buffer increased to 65536 bytes.

Check **http://www.marshallsoft.com** for the latest version of our FTP software.