FTP Client Engine
Library for Delphi

Programmer's Manual

(FCE4D)

Version 4.0

January 15, 2024

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

MarshallSoft Computing, Inc.
Huntsville AL 35815

Voice: 1.256.881.4630
Web: http://www.marshallsoft.com

MARSHALLSOFT is a registered trademark of MarshallSoft Computing.

TABLE OF CONTENTS

1 Introduction

The **FTP Client Engine for Delphi (FCE4D)** library is a toolkit that allows software developers to quickly develop FTP client applications in Delphi and Delphi for .NET.

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 Delphi FTP 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.

This **FTP Client Engine Programmers Manual** provides information need to compile and run programs in a Delphi programming environment.

The **FTP Client Engine for Delphi** component library supports and has been tested with all 32-bit and 64-bit versions of Delphi including:

- Borland Delphi (2.0, 3.0, 4.0, 5.0. 6.0 and 7.0)
- Borland Delphi 8 for .NET
- Borland Delphi 2005 & 2006
- Borland Turbo Delphi
- Codegear Delphi 2007
- Embarcadero Delphi 2009 & 2010
- Embarcadero Delphi XE, XE2, XE3, XE4, XE5,XE6,XE7,XE8
- Embarcadero Delphi 10 Seattle

FCE4D includes numerous example programs that demonstrate FTP processing to help software developers easily build software applications using FCE4D.

FCE4D works with all versions of 32-bit and 64-bit Windows through Windows 10. The **FTP Client Engine SDK** DLL's (FCE32.DLL and FCE64.DLL) can also be used from any development environment (Visual Basic, C++, Visual FoxPro, COBOL, Xbase++, dBase, PowerBASIC, etc.) capable of calling the Windows API.

When comparing **FTP Client Engine** component library against our competition, note that:

- FCE4D is a standard Windows DLL (NOT an OCX or ActiveX control) and is much smaller than a comparable OCX or ActiveX control.
- FCE4D does NOT depend on ActiveX or Microsoft Foundation Class (MFC) libraries or similar "support" libraries.
- FCE4D 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), PowerBASIC (FCE4PB), Visual FoxPro (FCE4FP), Visual dBASE (FCE4DB), and Xbase++ (FCE4XB). All versions of FCE use the same DLLs (FCE32.DLL or FCE64.DLL).

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:

- Works with 32-bit and 64-bit Windows through Windows 11.
- 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 Borland /CodeGear Delphi including Delphi 2005 through 2007, Delphi for
- .NET, Embarcadero Delphi 2009 - 2010, Embarcadero Delphi XE through XE8, Delphi 10 Seattle.
- Works with Delphi Prism for the .NET framework.
- Does **not** depend on support libraries. Makes calls to core Windows API functions <u>only</u>.
- Can be used with any program (in any language) capable of calling Windows API functions such   as Visual C++, .NET, Visual FoxPro, Visual Basic, VB.NET, Xbase++, dBASE, COBOL, Access and Excel.
- 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 (FCE_4D) in the set.

- FCE4D Programmer's Manual   (FCE_4D.PDF)
- FCE User's Manual   (FCE_USR.PDF)
- FCE Reference Manual   (FCE_REF.PDF)

The FCE_4D Programmer's Manual (FCE_4D.PDF) is the language specific (Delphi) 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 includes purchasing, licensing and technical support information.

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/fce4d.htm

1.3 Example Program

The following example demonstrates the use of some of the library functions using the FCE32.PAS unit.

```
var
   Code : Integer;
   Host : AnsiString;
   User : AnsiString;
   Pass : AnsiString;
begin
   {attach FCE}
   Code := fceAttach(1, FCE_KEY_CODE)
   if Code < 0 then
     begin
       {Error attaching FCE}
       exit
     end;
   {specify login parameters}
   Host : = 'ftp.drivehq.com' + Chr(0);
   User := 'anonymous' + Chr(0);
   Pass := 'you@your_isp.com' + Chr(0);
   {connect to FTP server}
   Code := fceConnect(0,@Host[1],@User[1],@Pass[1]);
   if Code < 0 then
     begin
       {Error code returned}
       exit
     end;
   {quit}
   fClose(0);
   fRelease()
end;
```

In the example program above, **fceConnect** is called to connect to the FTP server as user "anonymous" and password "you@your_isp.com". 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.

1.4 Installation

(1) Before installation of FCE4D, a Delphi compiler should already be installed on your system and tested. Note that Delphi 2 (or above) is required in order to create Win32 programs.

(2) Unzip FCE4D40.zip (evaluation version) or FCExxxxx.ZIP (registered version; xxxxx is the Customer ID).

(3) Run the installation program, SETUP.EXE, which will install all FCE4D files, including copying FCE32.DLL (and FCE64.DLL) to the Windows directory.

(4) For a quick start, load project file VER_PRJ.DPR

1.5 Uninstalling

Uninstalling FCE4D is very easy.

First, run UINSTALL.BAT, which will delete FCE32.DLL and FCE64.DLL from your Windows directory, which is normally C:\WINDOWS.

Second, delete the FCE project directory created when installing FCE4D.

1.6 Pricing

A developer's license for FCE4D 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).

1.7 Updates

When a developer license is purchased, 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's for a period of <u>one year</u> 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 /FCE4D/DOCS directory for more information.

2 Library Overview

The **FTP Client Engine Library for Delphi** has been tested on multiple computers running Windows XP through Windows 10.

The FCE4D library works with Borland/CodeGear/ Embarcadero Delphi including Delphi 2 – Delphi 10, Delphi 2005 – Delphi 2010, Turbo Delphi, Embarcadero Delphi XE – XE8, and Delphi 10 Seattle. FCE functions may be called by any Windows application program capable of calling the Windows API provided that the proper declaration file is used. Win32 and Win64 DLLs are provided.

The SETUP installation program will copy the FCE DLLs to the Windows directory. Refer to Section 1.4 "Installation". After SETUP is run, the FCE4D files are copied to the directory specified (default \FCE4D). Three sub-directories are created, as follows:

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

2.1 Dynamic Link Libraries

The **FTP Client Engine** library is implemented as a dynamic link library (DLL). 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".

The following files can be found in the DLL sub-directory when SETUP is run:

```
fce32.dll   - Win32 version of FCE
fce64.dll   - Win64 version of FCE
```

2.2 Keycode

FCE32.DLL and FCE64.DLL have a keycode encoded within them. The keycode is a 9 or 10 digit decimal number (unless it is 0), and will be found in the file KEYCODE.PAS. The keycode for the evaluation version is 0. The developer will receive a new key code after registering. SETUP copies the keycode to the FCE4D /APPS sub-directory (folder).

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 FCE32.DLL and FCE64.DLL from the Windows search.

2.3 Using the FCE Unit

The FTP Client library is written in ANSI C (like Windows itself). In C, strings are zero terminated. Note the manner in which strings are passed to FCE functions.

```
var
   Text :
begin
   ...
   {terminate strings passed to FCE with Chr(0)}
   HostStr := AnsiString(eServer.Text) + Chr(0);
   UserStr := AnsiString(eUser.Text) + Chr(0);
   PassStr := AnsiString(ePass.Text) + Chr(0);
   {connect to FTP server};
   Code := fceConnect(0, @HostStr[1], @UserStr[1], @PassStr[1]);
```

2.4 Adding FCE4D to A Project

Copy FCE32.PAS (or FCE64.PAS) to the same directory (folder) as the application program to which you want to add FCE code. You will find these files in the APPS directory (folder) created when you ran SETUP, usually C:\ FCE4D\APPS.

For Win32 Delphi, add `fce32` (fce32uc for Delphi .NET) and `keycode` to your "uses" clause in your source program (*.PAS). For example,

```
   uses
     fce32, keycode, ...
```

You can leave 'keycode' out above if you put your numerical keycode value (found in `keycode.pas`) directly into the call to `fceAttach`. Also add `fce32` to your project file (*.DPR). For example,

```
   uses
     fce32 in 'fce32.pas', ...

   {pass the key code}

   Code := fAttach(1, FCE_KEY_CODE)
```

2.5 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

3.0 Compiler Issues

The **FTP Client Engine for Delphi** component library supports all versions of Borland/CodeGear/Embarcadero Delphi for Win32 and Win64 as follows:

- Borland Delphi 2, 3, 4, 5, 6, 7 and 8.
- Borland Delphi 2005 (Delphi 9)
- Borland Delphi 2006
- Borland Turbo Delphi
- CodeGear Delphi 2007
- CodeGear Delphi 2009
- CodeGear Delphi 2010
- Embarcadero Delphi XE
- Embarcadero Delphi XE2 through XE8 (Win32 & Win64)
- Embarcadero Delphi 10 Seattle (Win32 & Win64)

3.1 Delphi Versions

Applications written with Delphi link with the same identical DLL's as for applications written in all other supported languages, such as C/C++ and Visual Basic.

Beginning with Delphi 2003, Delphi has two "personalities": (1) Win32 Delphi and (2) Delphi for .NET. Win32 Delphi programs are a continuation of the Delphi language as seen in earlier versions of Delphi. Delphi .NET is a version of Delphi designed to use the Microsoft .NET Framework.

3.1.1 Delphi 1

The first release of Borland Delphi (version 1) generated Win16 code. FCE4VB no longer supports 16-bit programs (VB3).

3.1.2 Delphi 2

Delphi version 2 and above generates Win32 code. Therefore, applications written using Delphi 2 will work with FCE32.DLL. Strings can be up to 2GB rather than 255 bytes as in Delphi 1.

Delphi 2 seems to have a problem with some of the PChar string functions. Although the default is "large strings", some of the string functions (such as StrPCopy) copy only 255 bytes. The MYSTRING.PAS unit contains a replacement unit to use instead of StrPCopy.

3.1.3 Delphi 3

Delphi 3 also has some problems with PChar string functions such as StrPCopy. See above section.

3.1.4 Delphi 4, 5 and 6.

There are no known Delphi problems impacting our example programs in Delphi 4 through Delphi 6. Application programs written using Delphi 4 through Delphi 6 will use FCE32.DLL.

3.1.5 Delphi 7 and 8.

Beginning in Delphi 7, the filename of a unit must match the unit name. Delphi 8 was a .NET only release.

3.1.6 Delphi 2005, Delphi 2006, and Delphi 2007

The Delphi 2005/2006/2007 compilers support both Win32 and the Microsoft .NET Framework.

When loading Delphi for Win32 projects with Delphi 2005/2006/2007, a window entitled "Project Upgrade" will be displayed:

This project must be upgraded before it can be opened. Please select which project type you wish to target:

```
( ) Delphi for .NET
( ) Delphi for Win32
```

Choose "Delphi for Win32" for all projects except "*.bdsproj" projects, which are Delphi for .NET projects.

3.1.7 Delphi 2009, Delphi 2010 and Delphi XE

In Delphi 2009 the definition of PChar was changed from a pointer to a 8-bit character to a pointer to a 16-bit character, also known as a "wide character". For this reason PAnsiChar must be used rather than PChar for pointers to buffers that are passed to FCE functions. Refer to the Win32 example programs in the APPS directory.

3.1.8 Delphi XE2 through XE8, Delphi 10 Seattle

Embarcadero Delphi XE2- XE10 can create both 32-bit and 64-bit executables. Note that although a 32-bit executable can run on both 32-bit and 64-bit Windows machines, a 64-bit executable can run only on a 64-bit Windows machine.

Step 1: Edit Files

1. Open file *_PGM.PAS using any text editor.
2. Replace FCE32 with FCE64 in the "uses" clause.
3. Open file *_PRJ.DPR using any text editor.
4. Replace FCE32 with FCE64.

Step 2: Delete Win32 files

1. Delete all *.DCU files that may have been created under Win32 Delphi.
2. Delete all *.DPROJ files that may have been created under Win32 Delphi.

Step 3: Change Win32 to Win64 in Delphi XE2 – XE10 Project Manager.

Start Delphi XE2 – XE10, load the project *_PRJ.DPR, then open the Project Manager found under "View" / "Project Manager" on the Delphi menu bar.

1. Right click on "Target Platforms (Win32)" and click on "Add Platform...".
2. When the "Select Platform" window is displayed, click "64-bit Windows".
3. Click [+] box to left of "Target Platforms (Win32)"
4. Right click on "32-bit Windows", then click "Remove Platform"

Note: The above changes have been made to the example programs. Unzip file **X64ZIP**

3.2 Compiling Example Programs

The example programs are compiled from the Delphi development environment using the provided Delphi project files (*.DPR). Recall that Delphi 1 generates Win16 applications while Delphi 2 and above generates Win32 (or Win64) applications.

FCE4D may also be used with "Borland Pascal for Windows".

3.3 Compiling FCE Source

The source code for the FCE DLL is written in standard ANSI C (FCE32.C), and has been compiled using Microsoft Visual C++. The Win32 version is compiled with the STDCALL and DECLSPEC compiler keywords. Source code for the FCE library can be purchased at the same time as a FCE developer license is purchased.

FCE may also be compiled using Borland C/C++ or Watcom C/C++ compilers. If you recompile FCE32.C using Borland or Watcom compilers, the resulting DLL can only be used by applications compiled with the same compiler, unless the STDCALL and DECLSPEC keywords are specified.

For more information on the C/C++ version of FCE, download the latest version of FCE4C from our web site at

http://www.marshallsoft.com/fce4c.htm

4 Example Programs

Some of the example programs use the DISPLAY.PAS, which contains 4 procedures:

```
DisplayChar   : Displays character.
DisplayString : Displays string.
DisplayLine   : Displays line.
DisplayError  : Displays error message.
DisplayPChar  : Displays PChar characters.
```

Delphi for .NET projects must be compiled with Delphi for .NET 2005/2006/2007 (or Delphi Prism).

Several example programs are included in **FTP Client Engine for Delphi**. Before writing your own programs, compile and run the example programs.

Compile and run the VER_PGM example first, in order to verify that you have installed FCE4D correctly and that you can use the FCE DLL (FCE32.DLL).

4.1 VER_PGM

The VER_PGM (Version Program) example program displays the FCE library version number and registration string. A TCP/IP connection is not required.

The project files are:

```
VER_PRJ.DPR : Project file.
VER_PGM.PAS : Program file.
VER_PGM.DFM : Delphi Form file.
```

4.2 LIST_PGM

The LIST_PGM example program connects to an FTP server and displays a listing of all files in the FTP server root directory.

The project files are:

```
LIST_PRJ.DPR : Project file.
LIST_PGM.PAS : Program file.
LIST_PGM.DFM : Delphi Form file.
```

4.3 FLD_PGM

The FLD_PGM example program is the same as the LIST_PGM example, except that it lists each line (in the listing of all files) by field.

The project files are:

```
FLD_PRJ.DPR : Project file.
FLD_PGM.PAS : Program file.
FLD_PGM.DFM : Delphi Form file.
```

4.4 GET_PGM

The GET_PGM example program connects to the MarshallSoft FTP server at ftp://ftp.marshallsoft.com and downloads all files (from the server directory) that match a user specified file specification using ? and * wildcard characters.

The project files are:

```
GET_PRJ.DPR : Project file.
GET_PGM.PAS : Program file.
GET_PGM.DFM : Delphi Form file.
```

4.5 PUT_PGM

The PUT_PGM example program connects to the MarshallSoft FTP server at ftp://ftp.marshallsoft.com and uploads all files (to the server directory) that match a user specified file specification using ? and * wildcard characters.

The project files are:

```
PUT_PRJ.DPR : Project file.
PUT_PGM.PAS : Program file.
PUT_PGM.DFM : Delphi Form file.
```

4.6 FTP_PGM

The FTP_PGM example program is a FTP application that can be used to connect to any FTP server and list, upload, download, and delete files, as well as navigate the server directory structure.

The project files are:

```
FTP_PRJ.DPR : Project file.
FTP_PGM.PAS : Program file.
FTP_PGM.DFM : Delphi Form file.
```

4.7 FTP2_PGM

The FTP2_PGM example program is the same program as FTP_PGM except that it operates in "direct mode" so that it can display the progress of uploads and downloads. Refer to the FCE User's Manual (FCE_USR) for more information on direct mode.

The project files are:

```
FTP2_PRJ.DPR : Project file.
FTP2_PGM.PAS : Program file.
FTP2_PGM.DFM : Delphi Form file.
```

4.8 GetNew

GetNew is a Delphi .NET (Prism) example project. It connects to the MarshallSoft FTP site at ftp://www.marshallsoft.com and downloads the file "fce-new.txt".

The Delphi 2005-2010 project files include:

```
GetNew_Project.* : Delphi 2005-2010 project files.
GetNew_WinForm.* : Delphi 2005-2010 source files.
fce32uc.pas      : Delphi 2005-2010 version of FCE declaration
file.
```

4.9 TIME

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

The project files are:

```
TIME_PRJ.DPR : Project file.
TIME_PGM.PAS : Program file.
TIME_PGM.DFM : Delphi Form file.
```

5 Revision History

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

Version 1.2: September 29, 1999

- The initial release of the Delphi version of FCE.

Version 2.0: June 5, 2000

- WriteBufferSize default reduced to 512.
- Added FCE_GET_LINE_COUNT to fceGetString.
- Rename file being downloaded by specifying "oldname:newname" for file name.
- Added FCE_SET_DATA_PORT to fceSetInteger.
- New example programs added.

Version 2.1: January 29, 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: November 8, 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.
- Modified FTPW to use proxy and download multiple files.

Version 2.3: January 10, 2003.

- Added FCE_SET_BLOCKING_MODE to control blocking (default ON) while connecting.
- Size of command buffer in fceCommand increased from 64 to 128 bytes.
- Added 200ms to minimum wait after sending password.
- Call control queue before each command.
- "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.
- Any keycode matches the shareware DLL.
- 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.
- 32 channels & 128 data ports supported.
- fceGetList returns error if receive buffer is too small.

Version 2.4: June 25, 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).

Version 2.5 August 1, 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 new version of GET_PRJ example program.
- Added fceShortToByte and fceByteToShort functions.
- Added support for Delphi 2005

Version 2.6: February 5, 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 15, 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 2, 2009

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

Version 3.1: July 11, 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: May 30, 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 28, 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().
- Added FCE_SET_DEBUG_LEVEL to fceSetInteger().
- Added support for Delphi XE4, Delphi XE5 and Delphi XE6.

Version 3.4:   November 24, 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.
- Tested with Delphi XE8 and Delphi 10 Seattle

Version 4.0: January 15, 2024

- 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/fce4vb.htm** for the latest version of our FTP client library software.