

FTP Client Engine Library for Delphi

Programmer's Manual

(FCE4D)

Version 3.1

July 11, 2011

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

Copyright (C) 2011
All rights reserved

MarshallSoft Computing, Inc.
Post Office Box 4543
Huntsville AL 35815

Voice : 1.256.881.4630
email : info@marshallsoft.com
web : www.marshallsoft.com

MARSHALLSOFT is a registered trademark of MarshallSoft Computing.

TABLE OF CONTENTS

| | | |
|------|----------------------------|---------|
| 1 | Introduction | Page 3 |
| 1.1 | Features | Page 4 |
| 1.2 | Documentation Set | Page 5 |
| 1.3 | Example Program | Page 5 |
| 1.4 | Installation | Page 6 |
| 1.5 | Uninstalling | Page 6 |
| 1.6 | Pricing | Page 6 |
| 1.7 | Updates | Page 6 |
| 2 | Library Overview | Page 7 |
| 2.1 | Dynamic Link Libraries | Page 7 |
| 2.2 | Keycode | Page 7 |
| 2.3 | Using the FCEW Unit | Page 7 |
| 2.4 | Using the FCE Unit | Page 8 |
| 2.5 | Adding FCE4D to a project | Page 8 |
| 2.6 | Error Display | Page 8 |
| 3 | Compiler Issues | Page 9 |
| 3.1 | Delphi Versions | Page 9 |
| 3.2 | 64-bit Delphi | Page 9 |
| 3.3 | Compiling Example Programs | Page 10 |
| 3.4 | Compiling FCE Source | Page 10 |
| 4 | Example Programs | Page 11 |
| 4.1 | VER_PGM | Page 11 |
| 4.2 | LIST_PGM | Page 11 |
| 4.3 | FLD_PGM | Page 11 |
| 4.4 | GET_PGM | Page 11 |
| 4.5 | PUT_PGM | Page 12 |
| 4.6 | FTP_PGM | Page 12 |
| 4.7 | FTP2_PGM | Page 12 |
| 4.8 | FTPW_PGM | Page 13 |
| 4.9 | SPD_PGM | Page 13 |
| 4.10 | GetNew | Page 14 |
| 4.11 | TIME | Page 14 |
| 5 | Revision History | Page 15 |

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 all versions of Borland (Codegear) Delphi including Delphi 2010 and .NET. **FCE4D** includes numerous example programs that demonstrate FTP processing to help software developers easily build software applications using FCE4D.

FCE4D runs under all versions of Windows (Windows 95, Windows 98, Windows ME, Windows 2000, Windows 2003, Windows NT, Windows XP, Windows Vista, Windows 7) and 64-bit Windows (Windows Vista and Windows 7). A Win32 DLL is provided with FCE4D (a Win64 DLL is available). 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:

- 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 Borland (Codegear) Delphi through Delphi 2010
 - Works with Delphi Prism for the .NET framework.
 - 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 FoxPro, Visual Basic, VB.NET, Xbase++, dBASE, COBOL, Access and Excel.
 - Supports Windows 95/98/Me/NT/2000/2003/XP/Vista.
-
- 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 and licensing 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 FCEW.PAS unit. The FCEW.PAS module is a "wrapper" unit for FCE.PAS (a copy of FCE32.PAS) that allows passing Delphi strings directly rather than having to first convert them to PCHAR variables. Refer to Section 2.4 below.

```
var
  Code : Integer;
begin
  {attach FCE}
  Code := fAttach(1, FCE_KEY_CODE)
  if Code < 0 then
    begin
      {Error attaching FCE}
      exit
    end;
  {connect to FTP server}
  Code :=
    fConnect(0, 'ftp.marshallsoft.com', 'anonymous', 'you@your_ism.com');
  if Code < 0 then
    begin
      {Error code returned}
      exit
    end;
  {quit}
  fClose(0);
  fRelease()
end;
```

In the example program above, **fConnect** is called to connect to the FTP server as user "anonymous" and password "you@your_ism.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. Access online at

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

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 FCE4D31.zip (evaluation version) or FCExxxx.ZIP (registered version; xxxx 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. No Windows system files are modified.

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

Note that the install process does not modify the Windows registry.

1.5 Uninstalling

Uninstalling FCE4D is very easy. FCE does not modify the registry or any Windows files.

First, run UINSTALL.BAT, which will delete FCE32.DLL from your Windows directory, typically C:\WINDOWS for Windows 95/98/Me/XP/2003/Vista or C:\WINNT for Windows NT/2000.

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

1.6 Pricing

A developer's license for FCE4D can be registered for \$115 (\$295 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). (http://www.marshallsoft.com/fce_usr.htm#Section_1.3)

1.7 Updates

When a developer license is purchased for FCE, the developer will be sent a registered DLL plus a license file (FCEExxxx.LIC). 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. A license can be updated for \$30 if ordered within one year of the original purchase (or previous update). Between one year and three years, licenses can be updated for \$55. After three years, updates are \$75.

Note that the registered DLL never expires.

Updates to the source code can be purchased for \$100 in addition to the cost of the update.

2 Library Overview

The **FTP Client Engine Library for Delphi** has been tested on multiple computers running Windows 95/98/Me/XP/Vista/Windows 7.

The FCE4D library works with all versions of 32-bit Delphi and Delphi for .NET. 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 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".

2.2 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.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 the FCE DLL from the Windows search.

2.3 Using the FCEW Unit

The FCEW.PAS module is a "wrapper" unit for FCE.PAS that allows you to pass Delphi strings directly rather than having to first convert them to PCHAR variables.

Compare the example program FTPW_PGM.PAS to FTP_PGM.PAS. Observe that FTPW_PGM.PAS has "uses fcew" while FTP_PGM.PAS has "uses fce".

Note that FCEW.PAS is easier to use while FCE.PAS has less overhead.

2.4 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 : String;
  Host : PChar;
begin
  {allocate Memory (128 bytes) and copy string to it}
  GetMem(Host, 128);
  StrPCopy(Host, 'ftp.marshallsoft.com');
  {pass Host to FCE function}
  Code := fceConnect(0,Host,...);
  {free Memory}
  FreeMem(Host, 128);
end;
```

FCE buffers can also be converted to Delphi strings with StrPas. For example:

```
Text := StrPas(Host);
```

Also see Section 2.3 "Using the FCEW Unit", in which the calls to GetMem and FreeMem are performed in FCEW.PAS rather than in your application.

2.5 Adding FCE4D to A Project

Copy FCE32.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.

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)
```

2.6 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

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.

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 through Delphi 7.

There are no known Delphi problems impacting our example programs in Delphi 4 through Delphi 7. Application programs written using Delphi 4 through Delphi 7 will use FCE32.DLL.

3.1.5 Delphi 2005 through Delphi 2010

Delphi 2010 is the latest version of (Codegear/Embarcadero) Delphi. Delphi Prism is the .NET development version of Delphi. Application programs written using Delphi 2005 through 2010 will use FCE32.DLL.

When loading Delphi 3 – Delphi 7 projects with Delphi 2005 – 2010, 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".

3.2 64-bit Delphi

FCE4D can support 64-bit code (Win64) if 64-bit Delphi is released by Embarcadero (previously Code Gear). Note that Microsoft Visual Studio supports Win64 as does FCE.

3.3 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 a Win16 application while Delphi 2 and above generates a Win32 application.

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

3.4 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

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 FTPW_PGM

The FTPW_PGM example program is the same program as FTP_PGM except that it calls functions in unit FCEW (which in turn call FCE functions) rather than FCE functions directly.

The FTPW_PGM program uses the FCEW.PAS unit as described in Section 2.4.

The project files are:

```
FTPW_PRJ.DPR : Project file.  
FTPW_PGM.PAS : Program file.  
FTPW_PGM.DFM : Delphi Form file.
```

Two additional features have been added to FTPW_PGM.

(1) FTPW_PGM is capable of connecting to a remote FTP server through a proxy server using the USER@SERVER proxy protocol. Check the "Proxy Host" check box, then enter the name or IP address of the proxy server, and the proxy port number (assigned by the particular proxy server).

Refer to the FCE User's Manual (FCE_USR) for a discussion of proxy servers and proxy protocols; Section 3.7 "Proxy Servers" and Section 3.8 "Proxy Protocols" of the FCE User's Manual (FCE_USR), (http://www.marshallsoft.com/fce_usr.htm#Section_3.7).

(2) FTPW_PGM can download multiple files using the wildcard pattern characters '?' and '*'. Enter the filename pattern into the pattern edit box, then select "Get Files".

4.9 SPD_PGM

The SPD_PGM 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.

The project files are:

```
SPD_PRJ.DPR : Project file.  
SPD_PGM.PAS : Program file.  
SPD_PGM.DFM : Delphi Form file.
```

4.10 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.
```

4.11 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

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

[END]