# FTP Client Engine Library

# for Visual Basic

# Programmer's Manual

**(FCE4VB)**

**Version 4.0**

**November 7, 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: http://www.marshallsoft.com

# TABLE OF CONTENTS

# 1 Introduction

The **FTP Client Engine for Visual Basic (FCE4VB)** library is a toolkit that allows software developers to quickly develop FTP client applications in Visual Basic, Visual Studio / VB.NET and VBA.

The **FTP Client Engine (FCE)** is a component 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 Visual Basic 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 Visual Basic programming environment.

The **FTP Client Engine for Visual Basic** component library supports and has been tested with all versions of Microsoft Visual Basic including Visual Basic .NET and Visual Studio. All programs will compile using VB-4.0 through VB-6.0 or Visual Studio. **FCE4VB** includes many Visual Basic examples.

FCE4VB runs under Windows XP 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, Xbase++, dBase, PowerBASIC, etc.) capable of calling the Windows API.

FCE4VB functions can also be called from Visual Basic for Applications (VBA) environments such as Excel, Microsoft Office, and Access. FCE4VB also works with Power Builder.

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

- FCE4VB is a standard Windows DLL (NOT an OCX or ActiveX control) and is much smaller than a comparable OCX or ActiveX control.
- FCE4VB does NOT depend on ActiveX or Microsoft Foundation Class (MFC) libraries or similar "support" libraries.
- FCE 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), Delphi (FCE4D), PowerBASIC (FCE4PB), Visual FoxPro (FCE4FP), Visual dBASE (FCE4DB), and Xbase++ (FCE4XB). 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 Microsoft Visual Basic (V4.0 through Visual Studio 2022)
- Can be used with Microsoft Visual Studio .NET
- Can be used with Visual Basic for Applications (VBA) programs and Power Builder.
- 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, Delphi, Xbase++, dBASE, COBOL, Access and Excel.
- Supports Windows XP through Windows 11.
- License covers all programming languages.
- Royalty free distribution with your compiled application.
- Can be purchased with or without source code
- 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_4VB.PDF) in the set.

- FCE4VB Programmer's Manual (FCE_4VB.PDF)
- FCE User's Manual (FCE_USR.PDF)
- FCE Reference Manual (FCE_REF.PDF)

The FCE_4VB Programmer's Manual (FCE_4VB.PDF) is the language specific (Visual Basic) 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 basic FTP processing as well as language independent programming issues such as application notes and includes purchasing and license 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/fce4vb.htm

## 1.3 Example Program

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

```
' 32-bit VB code segment
Dim Code As Long
' attach FCE
Code = fceAttach(1, FCE_KEY_CODE)
If Code < 0 Then
  Print "Error connecting"
  Exit Sub
End If
' connect to server
Code = fceConnect(0, "ftp.drivehq.com", "anonymous",
                  "you@yourisp.com")
' change to proper directory
Code = fceSetServerDir(0, "/MarshallSoft/PublicFolder")
' set to ASCII xfer mode
Code = fceSetMode(0, ASC("A"))
' download the file
Code = fceGetFile(0, "fce-new.txt")
' QUIT
Code = fceClose(0)
Code = fceRelease()
```

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

The server directory is changed to "pub/other", the transfer mode is set to ASCII, and the file "products.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.

## 1.4 Installation

(1) Before installation of FCE4VB, a Visual Basic compiler (VB4 through Visual Studio) should already be installed on your system and tested.

(2) Unzip FCE4VB40.ZIP (evaluation version) or FCExxxx.ZIP (registered version; xxxx is the Customer ID).

(3) Run the installation program, SETUP.EXE, which will install all FCE4VB files, including copying FCE32.DLL (and FCE64.DLL) to the Windows directory.

VB 4.0, VB 5.0 and VB 6.0 project filenames end with "32.vbp" and VB.NET and Visual Studio project filenames end with "vbproj." For example,

FCEVER32.VBP --- Project file for 32-bit Visual Basic (VB_4.0, 5.0, 6.0).
FCEVER.VBPROJ --- Project file for VB.NET / Visual Studio.
FCEVER(VS2008).VBPROJ --- Project file for Visual Studio 2008.
FCEVER(VS2008)x64.VBPROJ --- Project file for 64-bit Visual Studio 2008.
FCEVER(VS2010).VBPROJ --- Project file for Visual Studio 2010.
FCEVER(VS2010)x64.VBPROJ --- Project file for 64-bit Visual Studio 2010.
FCEVER(VS2012).VBPROJ --- Project file for Visual Studio 2012.
FCEVER(VS2012)x64.VBPROJ --- Project file for 64-bit Visual Studio 2012.
FCEVER(VS2013).VBPROJ --- Project file for Visual Studio 2013.
FCEVER(VS2013)x64.VBPROJ --- Project file for 64-bit Visual Studio 2013.
FCEVER(VS2015).VBPROJ --- Project file for Visual Studio 2015.
FCEVER(VS2015)x64.VBPROJ --- Project file for 64-bit Visual Studio 2015
FCEVER(VS2017).VBPROJ --- Project file for Visual Studio 2017.
FCEVER(VS2017)x64.VBPROJ --- Project file for 64-bit Visual Studio 2017.
FCEVER(VS2019).VBPROJ --- Project file for Visual Studio 2019.
FCEVER(VS2019)x64.VBPROJ --- Project file for 64-bit Visual Studio 2019.
FCEVER(VS2022).VBPROJ --- Project file for Visual Studio 2022.
FCEVER(VS2022)x64.VBPROJ --- Project file for 64-bit Visual Studio 2022.

## 1.5 Uninstalling

Uninstalling FCE4VB is very easy.

First, run UNINSTAL.BAT, which will delete FCE32.DLL and FCE64.DLL from the Windows directory, typically C:\WINDOWS.

Second, delete the FCE project directory created when FCE4VB was installed.

## 1.6 Pricing

A developer's license for FCE4VB can be purchased for $119 ($299 with ANSI C source code to the DLL). Purchasing details can be found in Section 1.3, "How to Purchase", of the FCE User's Manual (FCE_USR). Also see INVOICE.TXT or http://www.marshallsoft.com/order.htm

## 1.7 Updates

When a developer license is purchased for FCE, the developer will be sent registered DLLs plus a license file (FCEExxxx.LIC). The license file can be used to update the registered DLLs 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. The license can be updated for $33 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 $77. After 10 years, the update price is the same as the new price ($119). Note that the registered DLL's never expire.

# 2 Library Overview

The **FTP Client Engine Library for Visual Basic** has been tested on multiple computers running Windows XP through Windows 11.

The FCE4VB library has also been tested with several Visual Basic compilers, from VB_4.0 .0 through VB_6.0 .0 and Visual Studio through 2022. FCE can also be used with Power Builder as well as VBA applications such as Microsoft Office, EXCEL and ACCESS.

The SETUP installation program will copy the FCE DLLs to the Windows directory. Refer to Section 1.4 "Installation". After SETUP is run, the FCE4VB files are copied to the directory specified (default \FCE4VB). 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

The FCE DLLs 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.BAS (and KEYCODE.VB). The keycode for the evaluation version is 0. The developer will receive a new key code after purchasing a license. 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 purchasing, it is best to remove the evaluation version of the FCE DLLs from the Windows search.

## 2.3 Dynamic Strings

The Visual Basic language uses a technique known as "garbage collection" to manage string space at runtime and may be called internally at any time by the Visual Basic runtime, asynchronous to what you may be doing in your code.

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.

```
Dim Code As Integer
Dim Buffer As String * 80
' allocate buffer just before call to dllGetMessage
Buffer = SPACE$(80)
' copy message into 'Buffer'
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.

## 2.4 Error Display

The error message text associated with FCE error codes can be displayed by calling **fceErrorText**.

```
void DisplayError(int ErrCode, char *MsgPtr)
{int Len;
 char ErrBuff[129];
 printf("ERROR %d: ", ErrCode);
 if(MsgPtr) printf("%s: ", MsgPtr);
 Len = fceErrorText(ErrCode, (char *)ErrBuff, 128);
 if(Len>0) printf("%s\n", ErrBuff);
 else printf("\n");
}
```

## 2.5 FCE4VB Class

The FCE class "fceClass" (fceClass.cls) is a Visual Basic class wrapper for making calls to FCE32.DLL and FCE64.DLL The class name for each function is the same as the DLL function except the leading "fce" is replaced by "f".

Those functions that return strings do so by use of the "String Result" property. Instantiate **fceClass** as any other class in Visual Basic:

```
Dim C As New fceClass
```

Also see the FCE4VB Reference Manual (FCE_REF), the example project "HELLO32" and the file fceClass.txt.

Classes were added to Visual Basic beginning with version 5.0, and are required in order to compile **fceClass.**

## 2.6 Visual Studio .NET

There are a few differences between VB 4/5/6 and VB.NET/Microsoft Visual Studio that affect writing programs that use FCE.

(1) Variables that are declared “As Long” in VB_4/5/6 are declared “As Integer” in Visual Studio .Net.

(2) Fixed length strings are not supported in Visual Studio .Net. When calling any FCE function that can return a string, memory for the string variable must be allocated first. For example:

```
Buffer = Space(80)
Code = fceGetString(0, FCE_GET_REGISTRATION, Buffer, 80)
```

(3) Some VB functions must be fully qualified. For example, instead of `LEFT`, use `Microsoft.VisualBasic.Left`

(4) The module FCE32.VB or FCE64.VB (for 64-bit programs) must be included in all VB.Net programs. See the `HELLO.VB` example program.

## 2.7 Visual Basic for Applications (VBA)

The **FTP Client Engine** component library can be used with Microsoft VBA applications such as EXCEL, ACCESS, and Microsoft Office.

Start EXCEL (or other 32-bit Office VBA program such as MS Publisher, WORD or ACCESS), then enter design mode. Enable the "Controls Toolbox", choose "Tools" on the menu bar, then "Customize", and then check "Control Toolbox". From the control toolbox, choose and position a "Command Button". This will create code that looks like

```
Private Sub CommandButton1_Click()

End Sub
```

Replace the generated code with MODULE32.BAS. The easiest way to do this is to paste from the clipboard. Edit the 'HostName' in this code, using the name of the computer (or IP address in dotted decimal notation) where the server application (see SERVER.VBP) is running. Exit design mode and then press the command button to start this example program.

Also see VBA modules DNLOAD32.BAS (which downloads a file) and UPLOAD32.BAS (which uploads a file).

## 2.8 PowerBuilder

FCE can also be used with Power Builder applications. See PBUILDER.TXT for more information.

`FCE.PBI` : Power Builder declaration file.

## 2.9 Adding FCE4VB to A Project

Copy FCE32.BAS (if running VB 4/5/6), FCE32.VB (if running 32-bit VB.Net/Visual Studio), or FCE64.VB (if running 64-bit Visual Studio) 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:\FCE4VB\APPS.

### 2.9.1 Adding FCE4VB to a VB 4.0, 5.0, or 6.0 Project

Open an existing project with "File", "Open Project". Then choose "Insert", "Module", then add FCE32.BAS and KEYCODE.BAS to the project. If prompted to add "DAO 2.50 Object Library", choose "no". FCE functions can now be called from your VB program.

### 2.9.2 Adding FCE4VB to a Visual Studio / VB.Net Project

Open an existing project with "File", "Open Project". Then choose "Project", "Add Module", and then add FCE32.VB and KEYCODE.VB to the project. FCE functions can now be called from your Visual Studio / VB.NET program.

### 2.9.3 Adding FCE4VB to your VS 2005, VS 2008, VS 2010, VS 2012, …,VS 2022 Project

Open the existing project with "File", "Open Project". Then choose "Project", "Add Module", then add modules FCE32.VB (or FCE64.VB) and KEYCODE.VB to your project. If empty modules are created, replace the contents of these modules with the contents of the modules of the same name provided by us.

FCE functions can now be called from your Visual Studio VB program.

The error message text associated with FCE error codes can be displayed by calling **fceErrorText**.

```
void DisplayError(int ErrCode, char *MsgPtr)
{int Len;
 char ErrBuff[129];
 printf("ERROR %d: ", ErrCode);
 if(MsgPtr) printf("%s: ", MsgPtr);
 Len = FCEErrorText(ErrCode, (char *)ErrBuff, 128);
 if(Len>0) printf("%s\n", ErrBuff);
 else printf("\n");
}
```

# 3 Compiler Issues

The **FTP Client Engine for Visual Basic** component library supports and has been tested with all versions of 32-bit and 64-bit Microsoft Visual Basic (VB 4, VB 5, VB 6 and Microsoft Visual Studio VB.NET).

## 3.1 Visual Basic Makefiles

The first Visual Basic for Windows (version 3.0) uses a text file known as a "Visual Basic makefile" `(.MAK)` to list all the file components necessary to compile a program. Beginning in version 4.0, the "Visual Basic Project file" `(.VBP)` was added. Both formats are "project files".

## 3.2 Compiling Example Programs

Visual Basic project files (.VBP) are provided for all example VB 4/5/6 programs.

Visual Basic NET project files (.VBPROJ) are provided for the Visual Studio .Net example programs.

The example program code is stored inVB 4.0 (WIN32) format. Project files end with "32.VBP" (e.g.: FCEVER32.VBP). All 32-bit versions of VB (VB4 and above) can open files stored in VB 4.0 format.

When saving the example programs in VB version 5.0 or VB 6.0 format, answer "no" if asked to add the "Microsoft DAO v2.5 library".

Compile and run FCEVER32 as the first example. FCEVER does not require a TCP/IP connection.

## 3.3 Explicitly Loading a FCE DLL

When an application program runs that makes calls to FCE32.DLL or FCE64.DLL, the Windows operating system will locate FCE32.DLL (FCE642.DLL) by searching the directories as specified by the Windows search path. If the FCE32.DLL (FCE64.DLL) is placed in the \WINDOWS directory it will always be found by Windows.

FCE32.DLL or FCE64.DLL can be loaded from an explicit location by replacing “FCE32.DLL” in FCE32.BAS or FCE32.VB by the full path. For example, to load FCE32.DLL from C:\FCE4VB\APPS, the first entry in would be:

```
    Declare Function fceAbort Lib "C:\FCE4VB\APPS\FCE32.DLL" (ByVal channel As
Long) As Long
```

For Win64, substitute FCE64 for FCE32 in the above declaration.

## 3.4 Compiling FCE Source

FCE32.DLL is written in standard ANSI C (FCE32.C), and has been compiled using Microsoft Visual C/C++ with the STDCALL and DECLSPEC compiler keywords. Source code for the FCE library is provided for an additional cost. (See Section 1.6).

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 FCE4VB. The example programs are designed to demonstrate the various capabilities of FCE4VB. The best way to become familiar with FCE4VB is to study and run the example programs. Before writing your own programs, compile and run several of the example programs.

Each example program (except HELLO) comes with a VB project file. The HELLO program uses the VB class “fceClass”.

The example programs can be compiled with the version of Visual Basic indicated below. All example programs are located in the FCE4VB\APPS sub-directory.The first example program FCEVER displays the FCE library version number and registration string. It does not require a connection. Open project FCEVER32.VBP.

## 4.1 FCEVER

The program FCEVER (FCE Version) displays the FCE library version number and registration string.

The project files are:

```
FCEVER32.VBP  : For VB 4.0 and above.
FCEVER.VBPROJ : For Visual Studio (VB.NET).
FCEVER(VS2008).VBPROJ : For Visual Studio 2008.
FCEVER(VS2008)x64.VBPROJ : For 64-bit Visual Studio 2008.
FCEVER(VS2010).VBPROJ : For Visual Studio 2010.
FCEVER(VS2010)x64.VBPROJ : For 64-bit Visual Studio 2010.
FCEVER(VS2012).VBPROJ : For Visual Studio 2012.
FCEVER(VS2012)x64.VBPROJ : For 64-bit Visual Studio 2012.
... VS2013, VS2015, VS2017
FCEVER(VS2019).VBPROJ : For Visual Studio 2019.
FCEVER(VS2019)x64.VBPROJ : For 64-bit Visual Studio 2019.
FCEVER(VS2022).VBPROJ : For Visual Studio 2022.
FCEVER(VS2022)x64.VBPROJ : For 64-bit Visual Studio 2022.
```

## 4.2 GET

GET is an example FTP application that connects to our FTP server anonymously and downloads the file "fce-new.txt".

The project files are:

```
GET32.VBP  : For VB 4.0 and above.
GET.VBPROJ : For Visual Studio (VB.NET).
GET(VS2008).VBPROJ : For Visual Studio 2008.
GET(VS2008)x64.VBPROJ : For 64-bit Visual Studio 2008.
GET(VS2010).VBPROJ : For Visual Studio 2010.
GET(VS2010)x64.VBPROJ : For 64-bit Visual Studio 2010.
GET(VS2012).VBPROJ : For Visual Studio 2012.
GET(VS2012)x64.VBPROJ : For 64-bit Visual Studio 2012.
... VS2013, VS2015, VS2017
GET(VS2019).VBPROJ : For Visual Studio 2019.
GET(VS2019)x64.VBPROJ : For 64-bit Visual Studio 2019.
GET(VS2022).VBPROJ : For Visual Studio 2022.
GET(VS2022)x64.VBPROJ : For 64-bit Visual Studio 2022.
```

## 4.3 HELLO

The HELLO32 example program connects to a specified FTP server and verifies that it is responding to commands. HELLO32 uses the class **fceClass**, and therefore requires Visual Basic version 5.0 or higher.

The HELLO.VB program is the VB.NET equivalent of HELLO32. Also read section 2.6 "Visual Studio .NET" above.

The project files are:

```
HELLO32.VBP  : For VB 5.0 and above.
HELLO.VBPROJ : For Visual Studio (VB.NET).
HELLO(VS2008).VBPROJ : For Visual Studio 2008.
HELLO(VS2008)x64.VBPROJ : For 64-bit  Visual Studio 2008.
HELLO(VS2010).VBPROJ : For Visual Studio 2010.
HELLO(VS2010)x64.VBPROJ : For 64-bit  Visual Studio 2010.
HELLO(VS2012).VBPROJ : For Visual Studio 2012.
HELLO(VS2012)x64.VBPROJ : For 64-bit  Visual Studio 2012.
... VS2013, VS2015, VS2017
HELLO(VS2019).VBPROJ : For Visual Studio 2019.
HELLO(VS2019)x64.VBPROJ : For 64-bit  Visual Studio 2019.
HELLO(VS2022).VBPROJ : For Visual Studio 2022.
HELLO(VS2022)x64.VBPROJ : For 64-bit  Visual Studio 2022.
```

## 4.4 LIST

The LIST example program connects to a specified FTP server and lists all files by field. Before compiling, edit LIST32.FRM with the appropriate FTP parameters.

The project file is:

```
LIST32.VBP : for VB 4.0 and above.
```

## 4.5 MGET

The MGET example program downloads files according to a wildcard pattern using “?” and “*” characters. Also see the MPUT example program.

The project file is:

```
MGET32.VBP : for VB 4.0 and above.
```

## 4.6 MPUT

The MPUT example program uploads files according to a wildcard pattern using “?” and “*” characters. Also see the MGET example program.

The project file is:

```
MPUT32.VBP : for VB 4.0 and above.
```

## 4.7 FileTime

The FileTime example program requests the "File Modification Time" for a file. However, not all FTP servers support the MDTM command.

The project file is:

```
FileTime32.VBP : for VB 4.0 and above.
```

## 4.8 Module32

FCE4VB can be used with any VBA application. See section 2.7 “VBA Applications”

## 4.9 PROXY

The PROXY example program connects to a FTP server through a proxy server using the "USER@SERVER" protocol.

The project file is:

```
PROXY32.VBP : for VB 4.0 and above.
```

For a discussion of proxy servers and proxy protocols refer to the FCE User's Manual (FCE_USR.PDF), Section 3.7 “Proxy Servers" and Section 3.8 “Proxy Protocols”. The text file PROXY2.TXT in the \DOCS sub-directory also provides information.

## 4.10 SPEED

SPEED is a download speed test program. Note that the write buffer size is increased to 4096 (default is 1024) and the sleep time is reduced to 0 (default is 10 milliseconds):

```
Code = fceSetInteger(0, FCE_SET_WRITE_BUFSIZE, 4096)
Code = fceSetInteger(0, FCE_SET_SLEEP_TIME, 0)
```

The project file is:

```
SPEED32.VBP : for VB 4.0 and above.
```

## 4.11 WINFTP

WINFTP is an example FTP application that can be used to connect to a FTP server and upload, download, and delete files. WINFTP also demonstrates how to implement an upload/download progress bar.

The server, user, and password strings are read from the file WINFTP.INI when WINFTP begins execution, but can be changed at runtime before connecting to the FTP server.

The project files are:

```
WINFTP32.VBP  : For VB 4.0 and above.
WINFTP.VBPROJ : For Visual Studio (VB.NET).
WINFTP(VS2008).VBPROJ : For Visual Studio 2008.
WINFTP(VS2008)x64.VBPROJ : For 64-bit Visual Studio 2008.
WINFTP(VS2010).VBPROJ : For Visual Studio 2010.
WINFTP(VS2010)x64.VBPROJ : For 64-bit Visual Studio 2010.
WINFTP(VS2012).VBPROJ : For Visual Studio 2012.
WINFTP(VS2012)x64.VBPROJ : For 64-bit Visual Studio 2012.
... VS2013, VS2015, VS2017
WINFTP(VS2019).VBPROJ : For Visual Studio 2019.
WINFTP(VS2019)x64.VBPROJ : For 64-bit Visual Studio 2019.
WINFTP(VS2022).VBPROJ : For Visual Studio 2022.
WINFTP(VS2022)x64.VBPROJ : For 64-bit Visual Studio 2022.
```

## 5 Revision History

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

Version 1.2: August 16, 1999

Initial release of the Visual Basic version of FCE.

Version 2.0: May 3, 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.

Version 2.1: January 15, 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: September 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: November 26, 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 (random time bits no longer used).
- fceGetList returns error if receive buffer is too small.

Version 2.4: April 30, 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 Microsoft VB.NET example.

Version 2.5: July 22, 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.
- Added fceShortToByte and fceByteToShort functions.

Version 2.6: January 24, 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 10, 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: September 18, 2009

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

Version 3.1: July 8, 2011

- Log file is now time-stamped
- Added diagnostics to fceFileLength
- Function fceGetLocalFList no longer counts subdirectories
- Added fceGetFileSize function
- Added fceGetFileTime function
- Added Visual Studio 2010 support

Version 3.2: May 23, 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:  May 14, 2014

- Fixed problem with connecting w/o blocking.
- Fixed problem with server name being corrupted in the FCE log file.
- Added project files for Visual Studio 2013.
- Added project files for Visual Studio 2012.
- Added function fceGetSubDirs()
- Added FCE_SET_DEBUG_LEVEL to fceSetInteger().
- Added project files for VS 2012 and VS 2013.

Version 3.4:  October 30, 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 7, 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/fce4vb.htm** for the latest version of our FTP client library software.